Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

DIN EN ISO 15118-4

ICS 43.120

Einsprüche bis 2025-01-22
Vorgesehen als Ersatz für
DIN EN ISO 15118-4:2019-08;
Ersatz für
E DIN EN ISO 15118-4:2021-10

Entwurf

Straßenfahrzeuge – Kommunikationsschnittstelle zwischen Fahrzeug und Ladestation – Teil 4: Konformitätsprüfungen für das Netzwerk- und Anwendungsprotokoll (ISO/DIS 15118-4:2024); Englische Fassung prEN ISO 15118-4:2024

Road vehicles –
Vehicle to grid communication interface –
Part 4: Network and application protocol conformance test (ISO/DIS 15118-4:2024);
English version prEN ISO 15118-4:2024

Véhicules routiers –
Interface de communication entre véhicule et réseau électrique –
Partie 4: Essai de conformité du protocole d'application et du réseau
(ISO/DIS 15118-4:2024);
Version anglaise prEN ISO 15118-4:2024

Anwendungswarnvermerk

Dieser Entwurf mit Erscheinungsdatum 2024-11-22 wird der Öffentlichkeit zur Prüfung und Stellungnahme vorgelegt.

Weil das beabsichtigte Dokument von der vorliegenden Fassung abweichen kann, ist die Anwendung dieses Entwurfs besonders zu vereinbaren.

Stellungnahmen werden erbeten

— vorzugsweise online im Norm-Entwurfs-Portal von DIN unter www.din.de/go/entwuerfe bzw. für Norm-Entwürfe der DKE auch im Norm-Entwurfs-Portal der DKE unter www.entwuerfe.normenbibliothek.de, sofern dort wiedergegeben;

— oder als Datei per E-Mail an naautomobil@din.de möglichst in Form einer Tabelle. Die Vorlage dieser Tabelle kann im Internet unter www.din.de/go/stellungnahmen-norm-entwuerfe oder für Stellungnahmen zu Norm-Entwürfen der DKE unter www.dke.de/stellungnahme abgerufen werden;

— oder in Papierform an den DIN-Normenausschuss Auto und Mobilität (NAAutomobil), 10004 Berlin, Postfach 8 04 62 oder Behrenstr. 35, 10117 Berlin.

Es wird gebeten, mit den Kommentaren zu diesem Entwurf jegliche relevanten Patentrechte, die bekannt sind, mitzuteilen und unterstützende Dokumentationen zur Verfügung zu stellen.

Gesamtumfang 1628 Seiten

DIN-Normenausschuss Auto und Mobilität (NAAutomobil)

© DIN Deutsches Institut für Normung e. V. ist Inhaber aller einfachen Rechte der Verwertung, gleich in welcher Form und welchem Verfahren.
Alleinverkauf durch DIN Media GmbH, 10772 Berlin

www.din.de
www.dinmedia.de
3580567

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Nationales Vorwort

Das Dokument prEN ISO 15118-4:2024 wurde vom Technischen Komitee ISO/TC 22 „Road vehicles“ in Zusammenarbeit mit dem Technischen Komitee CEN/TC 301 „Straßenfahrzeuge“ erarbeitet, dessen Sekretariat von DIN (Deutschland) gehalten wird.

Das zuständige deutsche Normungsgremium ist der Arbeitsausschuss NA 052-00-31 AA „Datenkommunikation“ im DIN-Normenausschusses Auto und Mobilität (NAAutomobil).

Dieses Dokument enthält unter Berücksichtigung des DIN-Präsidialbeschlusses 1/2004 nur die Englische Fassung von prEN ISO 15118-4:2024.

Das DIN-Präsidium hat mit seinem Beschluss 1/2004 festgelegt, dass von dem in den Regeln der europäischen Normungsarbeit von CEN/CENELEC verankerten Grundsatz, wonach Europäische Normen in den drei offiziellen Sprachen Deutsch, Englisch und Französisch veröffentlicht werden, in begründeten Ausnahmefällen abgewichen und auf die deutsche Sprachfassung verzichtet werden kann.

Die Genehmigung dafür hat die DIN-Geschäftsleitung entsprechend ihren in Anlage 1 zu dem DIN-Rundschreiben A 5/2004 festgelegten Kriterien für die vorliegende Norm auf Antrag des NA 052 NAAutomobil als Ergebnis einer Einzelfallentscheidung erteilt, zumal bereits dieser Norm zugrunde liegende Papiere überwiegend in englischer Sprache auch von den deutschen Marktteilnehmern angewendet werden.

Um den Energieverbrauch von Fahrzeugen zu senken, werden Fahrzeuge mit elektrischem Teil- oder Komplettantrieb entwickelt. Um die Batterien dieser Fahrzeuge aufladen zu können, wird eine spezielle Ladeinfrastruktur benötigt.

Während verschiedene Teilaspekte in der Normung von Elektrofahrzeugen und Infrastruktur bei ISO und IEC bereits behandelt wurden, beschäftigt sich diese Normenreihe mit dem Informationsaustausch zwischen Elektrofahrzeug und Ladeinfrastruktur. Kommunikation ist für das effektive Aufladen von Fahrzeugen sowie die Entwicklung effizienter und komfortabler Abrechnungssysteme unabdingbar.

Dieser Teil der Normenreihe beschreibt spezifische Prüfungen basierend auf den Kommunikationsanforderungen, die in ISO 15118-2 beschrieben sind. Die Norm dient Systemlieferanten als Grundlage, um die Systemkonformität von Kommunikationssteuerungen entweder in Elektrofahrzeugen oder in Ladesäulen nachzuweisen.

Für die in diesem Dokument zitierten Dokumente wird im Folgenden auf die entsprechenden deutschen Dokumente hingewiesen:

IEC 61851-1:2017	siehe	DIN EN IEC 61851-1:2019-12
ISO 15118-1	siehe	DIN EN ISO 15118-1
ISO 15118-2:2014	siehe	DIN EN ISO 15118-2:2016-08
ISO 15118-3	siehe	DIN EN ISO 15118-3
ISO 15118-5	siehe	DIN EN ISO 15118-5

Aktuelle Informationen zu diesem Dokument können über die Internetseiten von DIN (www.din.de) durch eine Suche nach der Dokumentennummer aufgerufen werden.

Änderungen

Gegenüber DIN EN ISO 15118-4:2019-08 wurden folgende Änderungen vorgenommen:

a) Unterabschnitt 7.3.5 wurde gelöscht (Prüfüberwachung);

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

b) Qualität und Lesbarkeit der Bilder wurden gesteigert;

c) allgemeine Fehlerkorrekturen wurden im gesamten Dokument durchgeführt;

d) neue Abschnitte für Zuverlässigkeits- und Belastbarkeitsprüffälle wurden ergänzt (Abschnitt 11 und Anhang E).

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Nationaler Anhang NA
(informativ)

Begriffe

Reihenfolge und Inhalt der folgenden Begriffe sind identisch mit denen im Abschnitt Begriffe der Englischen Fassung.

Für die Anwendung dieses Dokuments gelten die Begriffe nach ISO 15118-1, ISO 15118-2, ISO 15118-3 und die folgenden Begriffe.

ISO und IEC stellen terminologische Datenbanken für die Verwendung in der Normung unter den folgenden Adressen bereit:

— ISO Online Browsing Platform: verfügbar unter https://www.iso.org/obp

— IEC Electropedia: verfügbar unter https://www.electropedia.org/

3.1
abstrakter Prüffall
en **abstract test case**
vollständige und unabhängige Spezifikation von Aktionen, die notwendig sind, um ein spezielles *Prüfziel* (3.24) sicherzustellen

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

Anmerkung 2 zum Begriff: Diese Spezifikation ist auf dem Abstraktionsniveau für eine bestimmte abstrakte Prüfmethode festgelegt, wobei in einem stabilen Prüfzustand gestartet und in einem stabilen Prüfzustand geendet wird und dabei eine oder mehrere nachfolgende oder gleichzeitige Verbindungen eingebunden sein können.

Anmerkung 3 zum Begriff: Die Spezifikation sollte in der Art und Weise vollständig sein, um ein Prüfurteil zu ermöglichen, welches eindeutig jedem potenziell überwachbaren Prüfergebnis zugeordnet werden kann (d. h. Reihe von Prüfereignissen).

Anmerkung 4 zum Begriff: Die Spezifikation sollte in der Art und Weise unabhängig sein, dass es möglich ist, den abgeleiteten ausführbaren Prüffall isoliert von anderen Prüffällen dieser Art auszuführen (d. h. die Spezifikation sollte immer die Möglichkeit des Startens und Beendens im Zustand „Leerlauf“ beinhalten).

3.2
abstrakte Prüfreihe
ATS, en: abstract test suite
Prüfreihe, die sich aus *abstrakten Prüffällen* (3.1) zusammensetzt, mit Bezug auf eine vorgegebene Anforderungsspezifikation

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

3.3
Black-Box-Prüfung
en **black box testing**
Prüfverfahren, welches das Verhalten eines *zu prüfenden Systems* (3.20) untersucht, ohne dabei die interne Implementierung und Struktur des *zu prüfenden Systems* zu berücksichtigen und sich somit auf die offenen Prüfschnittstellen am *zu prüfenden System* verlässt

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

3.4
Konformitätsanforderung
en **conformance requirement**
Anforderung an die Implementierung eines realen offenen Systems, von dem behauptet wird, dass es der/den relevanten Spezifikation(en) entspricht, die durch *Black-Box-Prüfungen* (3.3) validiert werden kann

Anmerkung 1 zum Begriff: Die Reihe von zusammenhängenden Konformitätsanforderungen legt das Verhalten des Systems und dessen Kommunikation fest. Konformität eines realen offenen Systems wird deshalb auf zwei Ebenen ausgedrückt: Konformität zu jeder individuellen Anforderung und Konformität zur Reihe. In diesem Dokument umfassen die anwendbaren Konformitätsprüfungen die Anforderungen, soweit sie durch *Black-Box-Prüfungen* (3.3) validiert werden können.

Anmerkung 2 zum Begriff: Siehe auch *statische Konformitätsanforderungen* (3.20) und *dynamische Konformitätsanforderungen* (3.6).

Anmerkung 3 zum Begriff: Die Anforderungen für diese Konformitätsspezifikation sind in ISO 15118-2 festgelegt.

3.5
normgerechte Implementierung
en **conforming implementation**
zu prüfendes System (3.20), das sowohl die *statischen Konformitätsanforderungen* (3.19) als auch die *dynamischen Konformitätsanforderungen* (3.6) erfüllt und das mit den in der *Erklärung zur Konformität der Protokollimplementierung* (3.16) angegebenen Fähigkeiten übereinstimmt

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

3.6
dynamische Konformitätsanforderung
en **dynamic conformance requirement**
Anforderung, die festlegt, welches überwachbare Verhalten nach der (den) relevanten Spezifikation(en) in Bezug auf die Kommunikation erlaubt ist

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

Anmerkung 2 zum Begriff: Die Anforderungen für diese Konformitätsspezifikation sind in ISO 15118-2 festgelegt.

3.7
ausführbarer Prüffall
en **executable test case**
Umsetzung eines *abstrakten Prüffalls* (3.1)

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

3.8
erwartetes Verhalten
en **expected behavior**
genaue Antwort des *SUT* (3.20), anhand der zugrunde liegenden Protokollspezifikation, auf den Stimulus, der im *Prüfverhalten* (3.21) festgelegt ist

3.9
Konformitätsaussage zur Implementierung
ICS, en: implementation conformance statement
Aussage des Lieferanten einer Implementierung oder eines Systems, von dem behauptet wird, dass es einer vorgegebenen Spezifikation entspricht, die ausdrückt, welche Fähigkeiten implementiert wurden

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Anmerkung 2 zum Begriff: Das vorgegebene Dokument für diese Konformitätsspezifikation ist ISO 15118-2.

3.10
zusätzliche Prüfinformation zur Implementierung
IXIT, en: implementation extra information for testing
Aussage, die von einem Lieferanten oder Umsetzer eines *zu prüfenden Systems* (3.20) getroffen wurde, die alle Informationen (zusätzlich zu den in der *ICS* (3.9) angegebenen) bezogen auf das *zu prüfende System* und dessen Testumgebung beinhaltet oder referenziert und es dem Prüflabor ermöglicht, eine geeignete Prüfreihe gegen das *SUT* durchzuführen

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

3.11
zu prüfende Implementierung
IUT, en: implementation under test
Implementierung eines oder mehrerer OSI-Protokolle in einer angrenzenden Nutzer-/Anbieter-Beziehung als Teil eines realen offenen Systems, welches durch die Prüfung untersucht wird

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

3.12
Hauptprüfkomponente
MTC, en: main test component
einzelne Prüfkomponente in einer Prüfkomponentenkonfiguration, verantwortlich für die Erstellung und Steuerung *paralleler Prüfkomponenten* (3.13) sowie für die Berechnung und Zuordnung des *Prüfurteils* (3.30)

Anmerkung 1 zum Begriff: Siehe ITU-T X.292.

3.13
parallele Prüfkomponente
PTC, en: parallel test component
Prüfkomponente, die von der *Hauptprüfkomponente* (3.12) erstellt wurde

Anmerkung 1 zum Begriff: Siehe ITU-T X.292.

3.14
Nachbedingung
en **post-condition**
Prüfschritte, die für die Festlegung des Weges vom Ende des *Prüfverhaltens (3.21)* bis zum finalen stabilen Zustand des *abstrakten Prüffalls* (3.1) benötigt werden

3.15
Vorbedingung
en **pre-condition**
Prüfschritte, die für die Festlegung des Weges vom beginnenden stabilen Zustand des *abstrakten Prüffalls* (3.1) bis zum initialen Zustand, an dem das *Prüfverhalten* (3.21) beginnt, benötigt werden

3.16
Erklärung zur Konformität der Protokollimplementierung
PICS, en: protocol implementation conformance statement
ICS (3.9) für eine Implementierung oder ein System, von dem behauptet wird, dass sie/es einer vorgegebenen Protokollspezifikation entspricht

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

Anmerkung 2 zum Begriff: Das vorgegebene Protokolldokument für diese Konformitätsspezifikation ist ISO 15118-2.

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Anmerkung 3 zum Begriff: Im Zusammenhang mit diesem Dokument spezifizieren PICS die Konfiguration des Prüfsystems (z. B. welche Prüffälle durchgeführt werden müssen und welche Konfiguration für jeden Prüffall erforderlich ist).

3.17
Zusatzinformation für die Prüfung der Protokollimplementierung
PIXIT, en: protocol implementation extra information for testing
IXIT (3.10), die auf die Konformitätsprüfung nach einer vorgegebenen Protokollspezifikation bezogen ist

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

Anmerkung 2 zum Begriff: Das vorgegebene Protokolldokument für diese Konformitätsspezifikation ist ISO 15118-2.

Anmerkung 3 zum Begriff: Im Zusammenhang mit diesem Dokument spezifizieren PIXITs die Konfiguration des *zu prüfenden Systems* (SUT) (z. B. welche Prüffälle durchgeführt werden müssen und welche Konfiguration für jeden Prüffall erforderlich ist).

3.18
Laufzeitumgebung
en **runtime environment**
Umgebung, die das Betriebssystem und entsprechende Plattform-Anforderungen eines Systems beschreibt

BEISPIEL Prüfsystem.

3.19
statische Konformitätsanforderung
en **static conformance requirement**
Anforderung, die festlegt, welche Limitierungen der zulässigen Kombinationen von implementierten Fähigkeiten in einem realen offenen System bestehen, welches mit der (den) relevanten Spezifikation(en) übereinstimmen soll

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

Anmerkung 2 zum Begriff: Das vorgegebene Protokolldokument für diese Konformitätsspezifikation ist ISO 15118-2.

3.20
zu prüfendes System
SUT, en: system under test
reales offenes System, das die *IUT* (3.11) beinhaltet

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

3.21
Prüfverhalten
en **test behavior**
Reihe von Prüfschritten (Prüfkörper), die notwendig sind, um den Prüfzweck sicherzustellen und um Urteile den möglichen Ergebnissen zuzuordnen

3.22
Prüfausführung
en **test execution**
Interpretation oder Ausführung einer *abstrakten Prüfreihe* (3.2)

Anmerkung 1 zum Begriff: Konzeptionell kann die Prüfausführung in drei interagierende Einheiten unterteilt werden: eine ausführbare Prüfreihe (ETS, en: executable test suite), einen Prüfrahmen (TFW, en: test framework) und ein optionales internes Kodierungs-/Dekodierungssystem (EDS, en: encoding/decoding system).

Anmerkung 2 zum Begriff: Siehe auch ETSI ES 201 873-5 V4.6.1.

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

3.23
Prüfrahmen
TFW, en: test framework
Einheit, die alle Aktionen von *ausführbaren Prüffällen* (3.7) oder Funktionen ausführt

Anmerkung 1 zum Begriff: Der Prüfrahmen interagiert mit den Einheiten Prüfsteuerung (TM, en: test management), SUT-Adapter (SA) und Programmadapter (PA) über die Prüfsteuerungsschnittstelle (TCI, en: test control interface) und die Prüflaufzeitschnittstelle (TRI, en: test runtime interface). Zusätzlich steuert der Prüfrahmen die ausführbare Prüfreihe (ETS) und das Kodierungs-/Dekodierungssystem (EDS). Diese Einheit führt alle notwendigen Aktionen aus, um einen Prüffall oder eine Funktion mit Parametern in der ETS-Einheit ordnungsgemäß auszuführen. Er befragt die Prüfsteuerungseinheit nach Modulparameterwerten, die für die ETS notwendig sind und sendet aufgezeichnete Informationen dorthin. Der Prüfrahmen sammelt und löst zugeordnete Urteile, die von der ETS-Einheit zurückgegeben werden.

Anmerkung 2 zum Begriff: Siehe auch ETSI ES 201 873-5 V4.6.1.

Anmerkung 3 zum Begriff: In diesem Dokument wird das Laufzeitsystem für den Prüfrahmen TTCN-3 verwendet, um eine Prüfrahmen-Funktionalität zu erläutern.

3.24
Prüfziel
en **test purpose**
Prosabeschreibung eines hinreichend festgelegten Prüfziels, welche sich auf eine einzelne *statische Konformitätsanforderung* (3.19) oder *dynamische Konformitätsanforderung* (3.6) oder eine Reihe konzentriert, wie in der entsprechenden OSI-Spezifikation festgelegt

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

BEISPIEL Überprüfung, ob ein spezifischer Wert eines spezifischen Parameters unterstützt wird.

3.25
Prüfsystem
en **test system**
reales System, welches den *Prüfrahmen* (3.23), die *abstrakte Prüfreihe* (3.2), die *Prüfausführung* (3.22) und die Adapter sowie die Codecs umfasst

Anmerkung 1 zum Begriff: Typischerweise beinhaltet es auch eine gemeinsame Laufzeitumgebung, basierend auf einem Betriebssystem.

3.26
Prüfssteuerungsschnittstelle
TCI, en: test control interface
vier Schnittstellen, welche die Interaktion der TTCN-3-Software mit der Prüfsteuerung, die Kodierung und Dekodierung, den Umgang mit den Prüfkomponenten und die Aufzeichnung in einem Prüfsystem festlegen

Anmerkung 1 zum Begriff: Siehe ETSI ES 201 873-6 V4.6.1.

3.27
Prüflaufzeitschnittstelle
TRI, en: test runtime interface
zwei Schnittstellen, welche die TTCN-3-Prüfsystem-Software zwischen dem *zu prüfenden System* (3.20) und dem Programmadapter (PA) sowie dem Systemadapter (SA) in einem *Prüfsystem* (3.25) festlegen

Anmerkung 1 zum Begriff: Siehe ETSI ES 201 873-6 V4.6.1.

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

3.28
Prüfsystemschnittstelle
TSI, en: test system interface
Prüfkomponente, welche die Zuordnung der im (abstrakten) TTCN-3-Prüfsystem verfügbaren Ports zu denjenigen bereitstellt, welche von einem realen Prüfsystem angeboten werden

Anmerkung 1 zum Begriff: Siehe ETSI ES 201 873-6 V4.6.1.

3.29
gültiges Prüfverhalten
VTB, en: valid test behavior
Prüfschritte, bei denen das Prüfsystem Stimuli an das *zu prüfende System* (3.20) sendet, die nach den Protokollanforderungen (syntaktisch und semantisch) gültig sind

Anmerkung 1 zum Begriff: Diese Art von Prüfverhalten ist in diesem Dokument festgelegt.

Anmerkung 2 zum Begriff: Die Protokollanforderungen für diese Konformitätsspezifikation sind in ISO 15118-2 festgelegt.

3.30
Urteil
Prüfurteil
en **verdict, test verdict**
Aussage „bestanden“, „fehlgeschlagen“ oder „ergebnislos“, wie in einem *abstrakten Prüffall* (3.1) festgelegt, hinsichtlich der Konformität eines *zu prüfenden Systems* (3.20) bezogen auf diesen Prüffall, wenn dieser gerade ausgeführt wird

Anmerkung 1 zum Begriff: Siehe ITU-T X.290.

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Nationaler Anhang NB
(informativ)

Literaturhinweise

DIN EN IEC 61851-1:2019-12, *Konduktive Ladesysteme für Elektrofahrzeuge — Teil 1: Allgemeine Anforderungen (IEC 61851-1:2017); Deutsche Fassung EN IEC 61851-1:2019*

DIN EN ISO 15118-1, *Straßenfahrzeuge — Kommunikationsschnittstelle zwischen Fahrzeug und Ladestation — Teil 1: Allgemeine Informationen und Festlegungen der Anwendungsfälle*

DIN EN ISO 15118-2:2016-08, *Straßenfahrzeuge — Kommunikationsschnittstelle zwischen Fahrzeug und Ladestation — Teil 2: Anforderungen an das Netzwerk- und Anwendungsprotokoll (ISO 15118-2:2014); Englische Fassung EN ISO 15118-2:2016, nur auf CD-ROM*

DIN EN ISO 15118-3, *Straßenfahrzeuge — Kommunikationsschnittstelle zwischen Fahrzeug und Ladestation — Teil 3: Anforderungen an physikalische- und Datenverbindungsschnittstelle*

DIN EN ISO 15118-5, *Straßenfahrzeuge — Kommunikationsschnittstelle zwischen Fahrzeug und Ladestation — Teil 5: Konformitätsprüfungen für die physikalische und Datenverbindungsschnittstelle*

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Contents

Page

1 Scope ... 1
2 Normative references ... 2
3 Terms and definitions ... 2
4 Abbreviated terms ... 7
5 Conventions ... 8
5.1 Requirement structure ... 8
5.2 Test system description ... 8
6 Test architecture reference model ... 8
6.1 General information ... 8
6.2 Platform adapter interface ... 9
6.3 SUT adapter interfaces ... 9
6.4 Codecs ... 12
7 Test suite conventions ... 12
7.1 General information ... 12
7.2 Test suite structure (TSS) ... 13
7.3 Test profiles ... 15
7.4 Test suite identifiers ... 23
7.5 Test suite coverage ... 29
7.6 Test case description ... 171
7.7 Test case specification ... 172
8 Test case descriptions for 15118-2 V2GTP ... 186
8.1 General information ... 186
8.2 SECC test cases ... 186
8.3 EVCC test cases ... 190
9 Test case descriptions for 15118-2 SDP messages ... 196
9.1 General information ... 196
9.2 SECC test cases ... 196
9.3 EVCC test cases ... 199
10 Test case descriptions for 15118-2 V2G application layer messages ... 234
10.1 General information ... 234
10.2 SECC test cases ... 234
10.3 EVCC test cases ... 412
11 Reliability and Robustness Testing ... 682
11.1 General information ... 682
11.2 SECC test cases ... 682
11.3 EVCC test cases ... 685
Annex A (normative) Configuration specifications ... 692
A.1 General information ... 692
A.2 Parameter configuration ... 692
A.3 Timer configuration ... 698
A.4 PICS configuration ... 699
A.5 PIXIT configuration ... 700
Annex B (normative) Test-case specifications for 15118-2 V2GTP ... 703
B.1 General information ... 703
B.2 SECC test cases ... 703
B.3 EVCC test cases ... 706

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex C (normative) Test-case specifications for 15118-2 SDP messages ... 711
C.1 General information ... 711
C.2 SECC test cases ... 711
C.3 EVCC test cases ... 713

Annex D (normative) Test-case specifications for 15118-2 V2G application layer messages ... 736
D.1 General information ... 736
D.2 SECC test cases ... 736
D.3 EVCC test cases ... 836

Annex E (normative) Test-case specifications for 15118-2 Reliability and Robustness Testing ... 1005
E.1 General information ... 1005
E.2 SECC test cases ... 1005
E.3 EVCC test cases ... 1009

Annex F (normative) Function specifications for supporting test execution ... 1017
F.1 General information ... 1017
F.2 Configuration functions ... 1017
F.3 Pre-condition functions ... 1019
F.4 Post-condition functions ... 1051
F.5 Common behavior functions ... 1053
F.6 Library functions ... 1058

Annex G (normative) Function specifications for 15118-2 V2GTP ... 1068
G.1 General information ... 1068
G.2 SECC functions ... 1068
G.3 EVCC functions ... 1070

Annex H (normative) Function specifications for 15118-2 SDP messages ... 1074
H.1 General information ... 1074
H.2 SECC functions ... 1074
H.3 EVCC functions ... 1076

Annex I (normative) Function specifications for 15118-2 V2G application layer messages ... 1080
I.1 General information ... 1080
I.2 SECC functions ... 1080
I.3 EVCC functions ... 1276

Annex J (normative) Function specifications for 15118-2 Reliability and Robustness Testing 1477
J.1 General information ... 1477
J.2 SECC functions ... 1477
J.3 EVCC functions ... 1483

Annex K (normative) Template specifications for V2G TCP/TLS Port Control ... 1510
K.1 General information ... 1510
K.2 V2G TCP/TLS port control templates ... 1510

Annex L (normative) Template specifications for 15118-2 V2GTP ... 1512
L.1 General information ... 1512
L.2 Common templates ... 1512

Annex M (normative) Template specifications for 15118-2 SDP messages ... 1514
M.1 General information ... 1514
M.2 Common templates ... 1514

Annex N (normative) Template specifications for 15118-2 V2G application layer messages ... 1515
N.1 General information ... 1515
N.2 Common templates ... 1515
N.3 SECC templates ... 1527
N.4 EVCC templates ... 1539

Annex O (normative) Template specifications for Security ... 1568
O.1 General information ... 1568
O.2 Common templates ... 1568

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex P (normative) Data type definitions **1573**
P.1 General information **1573**
P.2 Data types for PICS **1573**
P.3 Data types for PIXIT **1574**
P.4 Data types for V2G TCP/TLS Port Control **1576**
P.5 Data types for V2GTP **1576**
P.6 Data types for SDP messages **1577**
P.7 Data types for V2G messages **1578**
P.8 Data types for Security **1598**

Annex Q (informative) Overview of changed test cases compared to ISO 15118-4:2018 ..... **1600**

Bibliography **1612**

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 31, *Data communication*.

This second edition cancels and replaces the first edition (ISO 15118-4:2018), which has been technically revised.

The main changes are as follows:

— test control section removed to align with more atomic testing approach (7.3.5 in Ed. 1);

— quality and readability of all Figures improved;

— general error corrections and/or improvements for better consistency in test objective descriptions in Clauses 8, 9 and 10;

— general error corrections and /or improvements for better consistency in TTCN-3 test case specification in the Annexes;

— new clause on configuration of electrical parameters and power flow related data added (7.7.9)

— new clause on reliability and robustness test cases added (Clause 11 and Annex E);

— complete list of new, removed and updated test cases in this document is provided in Annex Q.

A list of all parts in the ISO 15118 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Introduction

The complexity resulting from the network and application protocol requirements specified in ISO 15118-2 for the Vehicle-to-Grid Communication Interface (V2G CI) requires a considerable amount of testing to enable interoperability between electric vehicles and charging stations. This document therefore specifies a conformance test suite for the network and application layer protocols of the V2G CI to derive a common and agreed basis for conformance tests. The resulting test suite is a prerequisite for downstream interoperability tests. Since interoperability furthermore involves the actual application logic of an implementation those tests are beyond the scope of this document (see NOTE 1). Hence this document focuses on the communication interface and the corresponding requirements given in ISO 15118-2 only.

NOTE 1 Practical limitations make it impossible to specify an exhaustive test suite, and economic considerations can restrict testing even further. Hence, the purpose of this document is to increase the probability that different implementations can interwork. This is achieved by verifying them by means of a protocol test suite, thereby increasing the confidence that each implementation conforms to the protocol specification. However, the specified protocol test suite cannot guarantee conformance to the specification since it detects errors rather than their absence. Thus, conformance to a test suite alone cannot guarantee interworking. What it does do is give confidence that a conforming implementation has the required capabilities and that its behavior conforms consistently in representative instances of communication.

NOTE 2 This document generally refers to system under test (SUT) instead of implementation under test (IUT), due to the black box testing paradigm adopted in this document and related certification processes.

NOTE 3 This document has some interdependencies to the conformance tests specified in ISO 15118-5 which result from ISO/OSI cross layer dependencies in the underlying protocol specification (e.g. for sleep mode).

NOTE 4 Even though ISO 15118-2:2014 references IEC 61851-1:2011 Edition 2, this document references IEC 61851-1:2017 Edition 3 because of applicability on the market.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Printed copies are uncontrolled

– Entwurf –

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Road vehicles — Vehicle to grid communication interface — Part 4: Network and application protocol conformance test

1 Scope

This document specifies conformance tests in the form of an abstract test suite (ATS) for a system under test (SUT) implementing an EVCC or SECC according to ISO 15118-2. These conformance tests specify the testing of capabilities and behaviors of an SUT as well as checking what is observed against the conformance requirements specified in ISO 15118-2 and against what the supplier states the SUT implementation's capabilities are.

The capability tests within the ATS check that the observable capabilities of the SUT are in accordance with the static conformance requirements specified in ISO 15118-2. The behavior tests of the ATS examine an implementation as thoroughly as is practical over the full range of dynamic conformance requirements specified in ISO 15118-2 and within the capabilities of the SUT.

A test architecture is described in correspondence to the ATS. The conformance test cases in this document are described leveraging this test architecture and are specified in TTCN-3 core language for ISO/OSI network layer (layer 3) and above. The conformance test cases for the Data Link Layer (layer 2) and physical layer (layer 1) are described in ISO 15118-5. Test cases with overlapping scopes are explicitly detailed.

This document does not include specific tests of other standards referenced within ISO 15118-2, e.g. IETF RFCs. Furthermore, the conformance tests specified in this document do not include the assessment of the performance of an implementation. They cannot provide judgments on the physical realization of abstract service primitives, how a system is implemented, how it provides any requested service, nor the environment of the protocol implementation. Furthermore, the abstract test cases specified in this document only consider the communication protocol specified ISO 15118-2. Power flow between the EVSE and the EV is no prerequisite for the test cases specified in this document.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 15118-1, *Road vehicles — Vehicle to grid communication interface — Part 1: General information and use-case definition*

ISO 15118-2:2014, *Road vehicles — Vehicle-to-Grid Communication Interface — Part 2: Network and application protocol requirements*

ISO 15118-3, *Road vehicles — Vehicle-to-Grid Communication Interface — Part 3: Physical and data link layer requirements*

ISO 15118-5, *Road vehicles — Vehicle to grid communication interface — Part 5: Physical layer and data link layer conformance test*

IEC 61851-1:2017, *Electric vehicle conductive charging system — Part 1: General requirements*

ETSI ES 201 873-5 V4.6.1, *TTCN-3: TTCN-3 Runtime Interface (June 2014)*

ETSI ES 201 873-6 V4.6.1, *TTCN-3: TTCN-3 Control Interface (June 2014)*

3 Terms and definitions

For the purpose of this document, the terms and definitions given in ISO 15118-1, ISO 15118-2, ISO 15118-3 and the following apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

3.1
abstract test case
complete and independent specification of the actions required to achieve a specific *test purpose* (3.24)

Note 1 to entry: See ITU-T X.290.

Note 2 to entry: This specification is defined at the level of abstraction of a particular abstract test method, starting in a stable testing state and ending in a stable testing state and can involve one or more consecutive or concurrent connections.

Note 3 to entry: The specification is complete in the sense that it is sufficient to enable a test verdict to be assigned unambiguously to each potentially observable test outcome (i.e. sequence of test events).

Note 4 to entry: The specification is independent in the sense that it is possible to execute the derived executable test case in isolation from other such test cases (i.e. the specification always includes the possibility of starting and finishing in the “idle” state).

3.2
abstract test suite
ATS
a collection of *abstract test cases* (3.1) with reference to a given requirement specification

Note 1 to entry: See ITU-T X.290.

 © ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

3.3
black box testing
method of testing that examines the behaviour of a *system under test* (3.20) without considering the internal implementation and structure of the *system under test*, thus relying on the *system under test's* open interface for testing

3.4
conformance requirement
requirement for implementation of a real open system which is claimed to conform to the relevant specification(s) that can be validated by *black box testing* (3.3)

Note 1 to entry: The set of conformance requirements together define the behavior of the system and its communication. Conformance of a real open system is, therefore, expressed at two levels, conformance to each individual requirement and conformance to the set. In this document, applicable conformance tests include requirements as far as they can be validated by *black box testing* (3.3).

Note 2 to entry: See also *static conformance requirements* (3.20) and *dynamic conformance requirements* (3.6).

Note 3 to entry: The requirements for this conformance specification are specified in ISO 15118-2.

3.5
conforming implementation
system under test (3.20) which satisfies both *static conformance requirements* (3.19) and *dynamic conformance requirements* (3.6), consistent with the capabilities stated in the *protocol implementation conformance statements* (3.16)

Note 1 to entry: See ITU-T X.290.

3.6
dynamic conformance requirement
requirement which specifies what observable behavior is permitted by the relevant specification(s) in instances of communication

Note 1 to entry: See ITU-T X.290.

Note 2 to entry: The requirements for this conformance specification are specified in ISO 15118-2.

3.7
executable test case
realization of an *abstract test case* (3.1)

Note 1 to entry: See ITU-T X.290.

3.8
expected behavior
exact response of the SUT (3.20) according to the underlying protocol specification to the stimulus defined in the *test behavior* (3.21)

3.9
implementation conformance statement
ICS
statement made by the supplier of an implementation or system claimed to conform to a given specification, stating which capabilities have been implemented

Note 1 to entry: See ITU-T X.290.

Note 2 to entry: The given document for this conformance specification is ISO 15118-2.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

3.10
implementation extra information for testing
IXIT
statement made by a supplier or implementer of a *system under test* (3.20) which contains or references all information [in addition to that given in the *implementation conformance statements* (3.9)] related to the *system under test* and its testing environment, which will enable the test laboratory to run an appropriate test suite against the SUT

Note 1 to entry: See ITU-T X.290.

3.11
implementation under test
IUT
implementation of one or more open system interconnection protocols in an adjacent user/provider relationship, being part of a real open system, which is to be studied by testing

Note 1 to entry: See ITU-T X.290.

3.12
main test component
MTC
single test component in a test component configuration responsible for creating and controlling *parallel test components* (3.13) and computing and assigning the *test verdict* (3.30)

Note 1 to entry: See ITU-T X.292.

3.13
parallel test component
PTC
test component created by the *main test component* (3.12)

Note 1 to entry: See ITU-T X.292.

3.14
post-condition
test steps needed to define the path from the end of the *test behavior* (3.21) up to the finishing stable state for the *abstract test case* (3.1)

3.15
pre-condition
test steps needed to define the path from the starting stable state of the *abstract test case* (3.1)up to the initial state from which the *test bevavior* (3.21) will start

3.16
protocol implementation conformance statement
PICS
implementation conformance statement (3.9) for an implementation or system claimed to conform to a given protocol specification

Note 1 to entry: See ITU-T X.290.

Note 2 to entry: The given protocol document for this conformance specification is ISO 15118-2.

Note 3 to entry: In the context of this document, PICS specify the configuration of the Test System (e.g. what test cases need to be executed and what confiugration is required for each testcase)

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

3.17
protocol implementation extra information for testing
PIXIT
implementation extra information for testing (3.10) related to testing for conformance to a given protocol specification

Note 1 to entry: See ITU-T X.290.

Note 2 to entry: The given protocol document for this conformance specification is ISO 15118-2.

Note 3 to entry: In the context of this document, PIXIT specify the configuration of the System under Test (SUT) (e.g. what protocol options are implemented and what confiugration is required for each testcase)

3.18
runtime environment
environment that describes the operating system and corresponding platform requirements of a system

EXAMPLE Test system.

3.19
static conformance requirement
requirement that specifies the limitations on the combinations of implemented capabilities permitted in a real open system which is claimed to conform to the relevant specification(s)

Note 1 to entry: See ITU-T X.290.

Note 2 to entry: The requirements for this conformance specification are specified in ISO 15118-2.

3.20
system under test
SUT
real open system in which the *implementation under test* (3.11) resides

Note 1 to entry: See ITU-T X.290.

3.21
test behavior
set of test steps (test body) which are essential to achieve the test purpose and assign verdicts to the possible outcomes

3.22
test execution
TE
interpretation or execution of an *abstract test suite* (3.2)

Note 1 to entry: Conceptually, the TE can be decomposed into three interacting entities: an Executable Test Suite (ETS), a Test Framework (TFW) and an optional internal Encoding/Decoding System (EDS) entity.

Note 2 to entry: See also ETSI ES 201 873-5 V4.6.1.

3.23
test framework
TFW
an entity to perform all actions of *executable test cases* (3.7) or functions

Note 1 to entry: The Test Framework interacts with the test management, system under test adapter and platform adapter entities via test control interface (TCI) and test runtime interface (TRI). It additionally initializes and manages the executable test suite and encoding/decoding system entities. This entity performs all the actions necessary to properly start the execution of a test case or function with parameters in the executable test suite entity. It queries the test management

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

entity for module parameter values required by the executable test suite and sends logging information to it. It also collects and resolves associated verdicts returned by the executable test suite entity.

Note 2 to entry: See also ETSI ES 201 873-5 V4.6.1.

Note 3 to entry: In this document, the test framework TTCN-3 runtime system is used to explain a test framework functionality.

3.24
test purpose
prose description of a well-defined objective of testing, focusing on a single *static conformance requirement* (3.19) or *dynamic conformance requirement* (3.6) or a set as specified in the appropriate OSI specification

Note 1 to entry: See ITU-T X.290.

EXAMPLE Verifying the support of a specific value of a specific parameter.

3.25
test system
real system combining the *test framework* (3.23), *abstract test suite* (3.2), *test execution* (3.22) and adapters as well as codecs

Note 1 to entry: Typically, also containing a common runtime environment based on an operating system.

3.26
test control interface
TCI
four interfaces that define the interaction of the TTCN-3 executable with the test management, the coding and decoding, the test component handling and the logging in a test system

Note 1 to entry: See ETSI ES 201 873-6 V4.6.1.

3.27
test runtime interface
TRI
two interfaces that define the interaction of the TTCN-3 executable between the system under test (3.20) and the platform adapter and the system adapter in a *test system* (3.25)

Note 1 to entry: See ETSI ES 201 873-6 V4.6.1.

3.28
test system interface
TSI
test component that provides a mapping of the ports available in the (abstract) TTCN-3 test system to those offered by a real test system

Note 1 to entry: See ETSI ES 201 873-6 V4.6.1.

3.29
valid test behavior
VTB
test steps where the test system sends stimuli to the *system under test* (3.20) that are valid (syntactically and semantically) according to the protocol requirements

Note 1 to entry: This type of test behavior is specified in this document.

Note 2 to entry: The protocol requirements for this conformance specification are specified in ISO 15118-2.

 © ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

3.30
verdict
test verdict
statement of “pass”, “fail” or “inconclusive”, as specified in an *abstract test case* (3.1), concerning conformance of an *system under test* (3.20) with respect to that test case when it is executed

Note 1 to entry: See ITU-T X.290.

4 Abbreviated terms

For the purpose of this document, the following abbreviations apply:

ALM	Application Layer Message
ATS	Abstract Test Suite
BEV	Battery Electric Vehicle
CA	Certificate Authority
CPL	Control Pilot Line
CRL	Certificate Revocation List
DH	Diffie Hellman
DER	Distinguished Encoding Rules
ECDSA	Elliptic Curve Digital Signature Algorithm
EDS	Encoding/Decoding System
EIM	External Identification Means
EMAID	E-Mobility Account Identifier
ETS	Executable Test Suite
EV	Electric Vehicle
EVCC	Electric Vehicle Communication Controller
EVSE	Electric Vehicle Supply Equipment
EXI	Efficient XML Interchange
HAL	Hardware Abstraction Layer
ICS	Implementation Conformance Statement
ID(s)	Identifier(s)
IUT	Implementation under Test
IXIT	Implementation eXtra Information for Testing
OCSP	Online Certificate Status Protocol
OEM	Original Equipment Manufacturer
PWM	Pulse Width Modulation
MTC	Main Test Component
NACK	Negative Acknowledgement
PDU	Protocol Data Unit
PICS	Protocol Implementation Conformance Statement
PIXIT	Protocol Implementation eXtra Information for Testing
PKI	Public Key Infrastructure
PLC	Power Line Communication
PnC	Plug and Charge
PTC	Parallel Test Component
SA	Secondary Actor
SDP	SECC Discovery Protocol
SECC	Supply Equipment Communication Controller

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

SLAC	Signal Level Attenuation Characterization
SUT	System Under Test
TC	Test Case (abstract or executable)
TCI	Test Control Interface
TCP	Transmission Control Protocol
TE	Test Execution
TFW	Test Framework
TP	Test Purpose
TRI	Test Runtime Interface
TSI	Test System Interface
TSS	Test Suite Structure
TTCN-3	Testing and Test Control Notation version 3
V2G	Vehicle-to-Grid
V2G CI	V2G Communication Interface
V2GTP	V2G Transfer Protocol
VTB	Valid Test Behavior
UDP	User Datagram Protocol
XML	Extensible Markup Language

5 Conventions

5.1 Requirement structure

This document uses unique number identifiers for each individual requirement. This requirement structure allows for easier requirement tracking and management. The following format is used throughout this document:

"[V2G"Y"-"XXX"]" requirement text

Where:

— **"**V2G**"** represents the ISO 15118 series,

— "Y" represents the document part of the ISO 15118 series, for this document Y = 4

— "XXX" represents the individual requirement number and

— "requirement text" includes the actual text of the requirement.

5.2 Test system description

TTCN-3 is used in this document to specify the test system and the abstract test cases in the abstract test suite. However, using TTCN-3 is not mandatory for the implementation of conformance tests of ISO 15118-2.

[V2G4-001] The implementers of conformance tests shall verify that all test objectives implemented in their environment are identical to those described in this document.

6 Test architecture reference model

6.1 General information

Figure 1 provides an overview on the test architecture for this document. Subclauses 6.2, 6.3 and 6.4 specify the interface requirements in Figure 1 for platform adapters, SUT adapters and all applicable codecs.

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Figure 1 — Test architecture reference model for testing of ISO 15118-2

For the scope of this document only the "ISO 15118-2 – Messaging" interface to the SUT and its dependencies to the corresponding "IEC 61851-1:2017, Annex A – PWM Signaling" interface are relevant. This document therefore limits its scope to the respective SUT adapters and codecs.

6.2 Platform adapter interface

The platform adapter within the test system is responsible for timers and external functions. Besides means for timers, which are typically provided as part of the test framework and runtime environment, there are external functions specified for XML security in 7.7.11 and EIM status control functionality in 7.7.12.

[V2G4-002] The platform adapter of the test system shall implement the TriPlatformPA and the TriPlatformTE interfaces as specified in ETSI ES 201 873-5 V4.6.1:2014, 6.5.3.

6.3 SUT adapter interfaces

6.3.1 General information

The SUT adapter within the test system adapts the TTCN-3 communication operations to the SUT based on an abstract test system interface and implements the real test system interface. It is responsible to propagate message requests and procedure-based calls from the test execution (see Figure 1) to the SUT, and to notify the test execution of any received test events by appending them to its port queues.

[V2G4-003] The SUT adapters of the test system shall implement the TriCommunicationSA and the TriCommunicationTE interfaces as specified in ETSI ES 201 873-5 V4.6.1:2014, 6.5.2.

NOTE 1 The actual implementation of these adapters is out of the scope of this document.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

6.3.2 IEC 61851-1 SUT adapter requirements

The IEC 61851-1 SUT adapter implements a parallel test component (PTC) for the control pilot and proximity pilot signalling specified in IEC 61851-1:2017 Annex A. It continuously controls and/or observes the positive voltage (state), frequency, duty cycle and the proximity resistor value during the entire test case execution in accordance with the specified valid range of values as specified in IEC 61851-1:2017, Annex A.

[V2G4-004] The IEC 61851-1 SUT adapter shall adapt the HAL 61851-1 functions to a PWM Signal as specified in 6.3.2.1 to 6.3.2.5.

6.3.2.1 Requirements of IEC 61851-1 SUT adapter for SECC or EVCC testing

[V2G4-022] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a function to get the current PWM state or associated voltage level.

[V2G4-023] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a call-back function which reports change events of the current PWM state or associated voltage level.

[V2G4-024] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a function to set the proximity on SUT side.

[V2G4-025] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a function to get the proximity resistor value from SUT side.

[V2G4-026] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a call-back function which reports change events of the resistor value for the proximity from SUT side.

[V2G4-027] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a function to get the duty cycle in [%].

[V2G4-028] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a call-back function which reports changes of the duty cycle in [%].

[V2G4-029] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a function to get the frequency in [Hz].

[V2G4-030] The IEC 61851-1 SUT adapter for SECC or EVCC testing shall provide a call-back function which reports change events of the frequency in [Hz].

6.3.2.2 Requirements of IEC 61851-1 SUT adapter for SECC testing

[V2G4-031] The IEC 61851-1 SUT adapter for SECC testing shall provide a function to set the current PWM state or associated voltage level. State B, C and D or the associated voltage level shall be supported, other voltage levels are optional.

6.3.2.3 Requirements of IEC 61851-1 SUT adapter for EVCC testing

[V2G4-032] The IEC 61851-1 SUT adapter for EVCC testing shall provide a function to set the current PWM state or associated voltage level. State E and F or the associated voltage level shall be supported, other voltage levels are optional.

[V2G4-033] The IEC 61851-1 SUT adapter for EVCC testing shall provide a function to set the current duty cycle in [%]. The values 5%, and 7%-98% shall be supported, other values are optional.

[V2G4-034] The IEC 61851-1 SUT adapter for EVCC testing shall provide a function to set current frequency in [Hz]. A value of 1000 Hz +/- 5% shall be supported; other values are optional.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

[V2G4-035] The IEC 61851-1 SUT adapter for EVCC testing shall provide a function to switch on/off the ControlPilot Line to simulate an un-plug or plug-in for the SUT. State A or the associated voltage level shall be supported, other voltage level is optional.

6.3.2.4 Requirements of IEC 61851-1 PTC for SECC or EVCC testing

[V2G4-036] The IEC 61851 PTC for SECC or EVCC testing shall provide a function which activates the validity monitoring as specified in [V2G4-037], [V2G4-038], [V2G4-039], [V2G4-040], [V2G4-041].

[V2G4-037] The IEC 61851-1 PTC for SECC or EVCC testing shall provide a function to confirm a change from/to a given PWM state. This function shall wait for receiving a state change event from IEC 61851-1 SUT adapter. If no change event occurs within a given time [ms], the function shall return an error.

[V2G4-038] The IEC 61851-1 PTC for SECC or EVCC testing shall provide a function to change its valid PWM state condition. This function shall change the internal monitoring condition to a given PWM state, which the IEC 61851-1 PTC will consider as the next valid PWM state.

[V2G4-039] The IEC 61851-1 PTC for SECC or EVCC testing shall provide a function to change its valid duty cycle range. This function shall change the internal monitoring condition to a given duty cycle range from [%] – to [%], which the IEC 61851-1 PTC will consider as the next valid duty cycle range.

6.3.2.5 Requirements of IEC 61851-1 PTC for SECC testing

[V2G4-040] The IEC 61851-1 PTC for SECC testing shall provide a function to confirm a given duty cycle in [%]. This function shall wait for receiving a duty cycle change event from IEC 61851-1 SUT adapter. If no change event occurs within a given time [ms], the function shall report an error.

[V2G4-041] The IEC 61851-1 PTC for SECC testing shall provide a function to change its valid frequency range. This function shall change the internal monitoring condition to a given frequency range from [Hz] – to [Hz], which the IEC 61851-1 PTC will consider as the next frequency range.

6.3.3 ISO 15118-2 SUT adapter requirements

[V2G4-005] The ISO 15118-2 SUT adapter shall adapt and send/receive the encoded SDP requests/responses through UDP messages to/from the SUT as specified in ISO 15118-2:2014, 7.7.2 and 7.10.1.

[V2G4-006] The ISO 15118-2 SUT adapter shall adapt and send/receive the encoded V2G messages through a TCP or TLS connection to/from the SUT as specified in ISO 15118-2:2014, 7.7.1 and 7.7.3.

[V2G4-007] The ISO 15118-2 SUT adapter shall adapt the internal port control messages for TCP and TLS connection management as specified in Annex P.4

NOTE 3 The TCP/TLS connection management includes the following functionalities:

- control for TCP/TLS port disconnection handling;
- control for receiving information / status on a given TCP/TLS connection ;
- notification of TCP/TLS connection status events;
- notification of TLS Handshake status events.

[V2G4-008] The ISO 15118-2 SUT adapter shall provide a port map operation (refer to 7.3.3) with additional parameters for TCP and TLS connection configuration as specified in Annex P.4.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

NOTE 4 The TCP/TLS connection configuration includes the following functionalities:

- configuration for IP address and port;
- configuration for usage of security (TCP/TLS) ;
- configuration for cipherSuite;
- configuration for TLS protocol version;
- configuration for TLS certificate.

NOTE 5 The ISO 15118-3 adapter is specified in ISO 15118-5.

6.4 Codecs

A codec is responsible for the external encoding and decoding of TTCN-3 values into bit strings suitable to be sent to the SUT. The test execution (TE) determines which codec is used and passes the TTCN-3 data to the appropriate encoder to obtain the encoded data. Received data is decoded in this entity by using the appropriate decoder, which translates the received data into TTCN-3 values cf. ETSI ES 201 873-6 V4.5.1, that can be matched against expected behaviors in the form of templates.

[V2G4-009] All codecs in this document shall implement the TCI-CD interface as specified in ETSI ES 201 873-6 V4.6.1:2014, 7.3.2.

NOTE 1 For ISO 15118-2 conformance testing, three codecs (V2G-EXI, SDP, V2GTP, see Figure 1) are used to encode or decode messages consumable by the test system into bit strings / signals consumable by the SUT.

NOTE 2 The exact implementation of these codecs is out of the scope of this document.

[V2G4-010] The V2G-EXI codec shall encode TTCN-3 values as specified in Annex P.7 into corresponding EXI streams as specified in ISO 15118-2:2014, 7.9.1.

[V2G4-011] The V2G-EXI Security codec shall encode TTCN-3 values for X.509 Certificates as specified in Annex P.7 into a corresponding Base64 representation suitable for EXI streams according to ISO 15118-2:2014, 7.9.1.

[V2G4-012] The V2G-EXI codec shall decode EXI streams as specified in ISO 15118-2:2014, 7.9.1 into TTCN-3 values as specified in Annex P.7.

[V2G4-013] The V2G-EXI Security codec shall decode Base64 representation from EXI streams according to ISO 15118-2:2014, 7.9.1 into TTCN-3 values for X.509 Certificates as specified in Annex P.7.

[V2G4-014] The SDP codec shall encode TTCN-3 values as specified in Annex P.6 into corresponding SDP requests/responses as specified in ISO 15118-2:2014, 7.10.1.3, 7.10.1.4 and 7.10.1.5.

[V2G4-015] The SDP codec shall decode SDP requests/responses as specified in ISO 15118-2:2014, 7.10.1.3, 7.10.1.4 and 7.10.1.5 into TTCN-3 values as specified in Annex P.5.

[V2G4-016] The V2GTP codec shall encode TTCN-3 values as specified in Annex P.5 into corresponding V2GTP envelopes as specified in ISO 15118-2:2014, 7.8.3.1.

[V2G4-017] The V2GTP codec shall decode V2GTP envelopes as specified in ISO 15118-2:2014, 7.8.3.1 into TTCN-3 values as specified in Annex P.5.

7 Test suite conventions

7.1 General information

This clause specifies all conventions that are relevant for conformance tests of SUTs implementing ISO 15118-2.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

7.2 Test suite structure (TSS)

An abstract test suite is a complete set of abstract test cases, possibly combined into groups or modules (e.g. for use cases or domains like AC, DC charging), that are necessary to perform conformance testing for a given SUT.

Each abstract test case specifies how the SUT is stimulated with specific inputs and furthermore specifies expected reaction(s) for verdict assignment. Depending on the test purpose different pre-conditions and post-conditions are considered for the formulation of the test behavior. The pre-conditions, post-conditions as well as test behaviors are encapsulated into individual functions and stored within separate modules. Thus, an abstract test case is composed by the actual test behavior enveloped by pre- and post-conditions (see 7.7.6 for detail). The corresponding grouping of functions can therefore be assigned to the lowest abstract hierarchical level (see Figure 2). The abstract test cases are assigned to the second level.

Figure 2 — General overview of the Test Suite Structure (TSS)

The test profile is a collection of self-contained test cases as well as PICS (see 7.3.4) and PIXIT (see 7.3.5) in order to represent a given use case. The selection is based on the use cases of the ISO 15118 series and its corresponding requirements.

Hence the abstract test suite structure (TSS) is segmented into subgroups according to the use cases specified in ISO 15118-1 for conformance testing. Table 1 shows these subgroups, which are used for the organization of the abstract test case specifications as well as for the test suite identifiers (see 7.4 for detail).

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table 1 — Identifiers within the Test Suite Structure (TSS)

Identifiers	Values	Description
`<sut>`		System under test
	`EVCC`	Electric Vehicle Communication Controller
	`SECC`	Supply Equipment Communication Controller
`<dom>`		Domain
	`AC`	AC specific behaviors
	`DC`	DC specific behaviors
	`IN`	Inductive specific behaviors
	`CMN`	Common behaviors
`<ctx>`	`{fullname}`	Context (e.g. name of message pattern according to ISO 15118-2)

NOTE The domain for inductive charging is not relevant for this document but is introduced for future purposes.

Table 2 describes the Annexes derived for this document.

Table 2 — Test suite structure – Annexes description

Annexes	Description
Configuration Annex A	The configuration annex contains constant definitions timers, PICS and PIXIT definitions. The configuraton described in Annex A applies.
Test cases Annex B - E	The test case annexes contain the TTCN-3 test case files. The EVCC subclauses contain the abstract test cases if the SUT is an EVCC. The SECC subclauses contain the abstract test cases if the SUT is an SECC. The abstract test cases in Annex B- Annex E apply.
Functions Annex F – J	The function annexes contain methods for supporting the test execution as well as the actual pre-, post-conditions and test behaviors of the ISO 15118 conformance tests. The EVCC subclauses contain the test behavior if the SUT is an EVCC. The SECC subclauses contain the test behavior if the SUT is an SECC. The pre-condition sub clause contains functions that are used for defining the pre-conditions of a test behavior. The post-condition subclause contains functions that are used to establish the post-conditions of a test case. The library functions subclause contains utility functions used across the test suite for various purposes. The functions in Annex F - Annex J apply.
Templates Annex K - O	The templates sub clause contains the TTCN-3 template files for matching a SUT's reaction on a stimulus against its expected behavior. The templates in Annex K - Annex O apply.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Data Structures Annex P	The data structures sub clause contains data structures that are needed additionally to the data structures of the ISO 15118 series, e.g. used for X.509 certificates. The data structures in Annex P apply.

7.3 Test profiles

7.3.1 General information

This sub clause specifies test profiles for conformance with ISO 15118-2. A test profile consists of a test configuration as well as a selection and assignment of PICS/PIXIT. Depending on the test configuration a set of test components and ports are specified. The test profile furthermore includes a test group defining the set of relevant test cases and the sequence in which they are executed in order to perform a conformance test for a given use case.

7.3.2 Test configurations

The test configuration reflects various ISO 15118 scenarios. Entities for the SUT, test system and PTCs are:

— Electric Vehicle Communication Controller (EVCC)

— Supply Equipment Communication Controller (SECC)

— [PWM Signaling (61851)]

Table 3 shows combinations of such entities defining valid test configurations for this document (CF_Part_ID).

Table 3 — ISO 15118-4 test configurations

CF_Part_ID	SUT	Test system	PTCs
CF_04_001	SECC	EVCC	PTC1=61851
CF_04_002	EVCC	SECC	PTC1=61851

7.3.3 Components and ports

In correspondence to the identified set of relevant test configurations this sub clause specifies test components which reflect the main entities needed for stimulation of the SUT with respect to ISO 15118. Ports are used to connect these components with each other and the SUT. The components are specified in the configuration module and are summarized in Table 4. Port types specify which kind of messages can be sent or received by this port. The corresponding port type definitions are summarized in Table 5.

Table 4 — Component definitions

Components	Description
`SECC_Tester` (MTC)	This component type is the main type for the tests of an SECC. A `V2G_TCP_TLS_ALM_SECC_Port` , `V2G_UDP_SDP_Port`, `HAL_61851_Port` and a `HAL_61851_Internal_Port` (see Table 5) are assigned to this component type.
`EVCC_Tester` (MTC)	This component type is the main type for the tests of an EVCC. A `V2G_TCP_TLS_ALM_EVCC_Port` , `V2G_UDP_SDP_Port`,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

	HAL_61851_Port and a HAL_61851_Internal_Port (see Table 5) are assigned to this component type.
HAL_61851_Listener (PTC)	The HAL_61851_Listener is responsible for watching the correct IEC 61851 behavior and is used in the form of a parallel test component (PTC). This component uses the HAL_61851_Port (see Table 5) to communicate with an IEC 61851-1 SUT Adapter and the HAL_61851_Internal_Port to communicate with the MTC.

Table 5 — Port type definitions

Port Type	Description
V2G_TCP_TLS_ALM_SECC_Port	This port is used to send and receive V2G- Application Layer Messages (ALM) or supportedAppProtocol messages via TCP/TLS specified in ISO 15118-2 to/from an SECC. This port is also used to send and receive internal TCP/TLS port control messages specified in Annex P.
V2G_TCP_TLS_ALM_EVCC_Port	This port is used to send and receive V2G- Application Layer Messages (ALM) or supportedAppProtocol messages via TCP/TLS specified in ISO 15118-2 to/from an EVCC. This port is also used to send and receive internal TCP/TLS port control messages specified in Annex P.
V2G_UDP_SDP_Port	This port is used to send and receive SDP messages via UDP specified in ISO 15118-2 to/from an EVCC or SECC.
HAL_61851_Port	This port is used to send or receive messages to an IEC 61851 communication unit. The communication unit will interpret these messages and will set the corresponding signaling to the SUT (SECC/EVCC). The specification of the IEC 61851 communication unit is out of the scope of this document.
HAL_61851_Internal_Port	This port is used to synchronize the status of the IEC 61851-1 PWM signal between the MTC (SECC_Tester or EVCC_Tester) and PTC (HAL_61851_Listener_Port). It enables the MTC and PTC to send and receive HAL 61851-1 messages.

These components and ports compose relevant test configurations for this document. Whether the type EVCC_Tester or SECC_Tester is used as MTC depends on the type of the SUT. If the SUT is an EVCC, the MTC uses the type EVCC_Tester. If the SUT is an SECC, the MTC uses the type SECC_Tester. The HAL_61851_Listener combines all necessary listener functionality into one component, independent on whether the MTC represents an SECC_Tester or an EVCC_Tester. The MTC always contains a TTCN-3 test configuration and delimits the lifeline during test execution. Next to using ports for communication purposes, local timers, variables or constants can be assigned to components to store dynamic information in a given executable test case.

In addition to the MTC and PTC and their corresponding port type definitions, a test configuration also consists of respective Test System Interfaces (TSI). An abstract Test System Interface (TSI) is specified as a collection of ports. A TSI has no local timers, constants or variables. Only ports are assigned to it. In a given executable

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

test case, test components ports can be mapped dynamically to the TSI ports to establish communication channel to the real test system interface. In the test configuration the TSI uses the type System_EVCC or System_SECC depending on the type of the SUT. If the SUT is an EVCC, the TSI uses the type System_EVCC. If the SUT is an SECC, the TSI uses the type System_SECC.

The test configuration is illustrated in Figure 3. The type of the V2G components and ports (EVCC or SECC) depends on the SUT type.

Figure 3 — ISO 15118-4 test configuration

As shown in Figure 3 the port mappings are specified statically as follows:

— The port `pt_V2G_TCP_TLS_ALM_EVCC_Port` of the TSI is always mapped to port `pt_V2G_TCP_TLS_ALM_EVCC_Port` of the MTC.

— The port pt_V2G_TCP_TLS_ALM_SECC_Port of the TSI is always mapped to port `pt_V2G_TCP_TLS_ALM_SECC_Port` of the MTC.

— The port `pt_V2G_UDP_SDP_Port` of the TSI is always mapped to port `pt_V2G_UDP_SDP_Port` of the MTC.

— The port `pt_HAL_61851_Port` of the TSI is always mapped to port `pt_HAL_61851_Port` of the MTC.

— The `port pt_HAL_61851_Listener_Port` of the TSI is always mapped to port `pt_HAL_61851_Listener_Port` of the `v_HAL_61851_Listener`. The `pt_HAL_61851_Listener_Port` is from the same type `HAL_61851_Port` as the `pt_HAL_61851_Port`.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

— The internal port `pt_HAL_61851_Internal_Port` is not mapped to the TSI. This port is connected between the MTC and PTC.

7.3.4 Protocol Implementation Conformance Statement (PICS) Definition

To evaluate the conformance of a particular implementation, it is necessary to have a statement of the capabilities and options which have been implemented, and any features which have been omitted, so that the implementation can be tested for conformance against relevant requirements, and against those requirements only. Those capabilities and features are typically referred to as Implementation Conformance Statements, ICS. In the context of a communication protocol as specified in ISO 15118-2 such statements are referred to as Protocol Implementation Conformance Statement (PICS) cf. [ITU-T X.290].

All PICS specified in the ATS are summarized in Table 6 - Table 8. These PICS specify selection criteria in the test system for an appropriate test case selection considering a given SUT. The PICS also specify appropriate configuration settings in the test system for each test case.

Table 6 — PICS for test system configurations CF_04_001 and CF_04_002 (SUT either SECC or EVCC)

PICS	Description
`PICS_CMN_CMN_CombinedTesting`	Indication for enabling combined testing including SLAC association & V2G messaging
`PICS_CMN_CMN_ChargingMode`	Indication for testing either AC or DC charging mode
`PICS_CMN_CMN_IdentificationMode`	Indication for testing either EIM or PnC identification mode
`PICS_CMN_CMN_PlugType`	Indication for testing either type1 or type2 plug type
`PICS_CMN_AC_CableCapability`	Indication for used AC cable type Choice: i) capability13A, ii) capability20A, iii) capability32A, iv) capability63A, v) capability70A NOTE: The choices map to the following Proximity resistor values: – capability13A corresponds to resistor value equal to 1500 Ohm – capability20A corresponds to resistor value equal to 680 Ohm – capability32A corresponds to resistor value equal to 220 Ohm – capability63A corresponds to resistor value equal to 100 Ohm (1-phase) – capability70A corresponds to resistor value equal to 100 Ohm (3-phase)
`PICS_CMN_CMN_WakeUp`	Indication for the pre-defined wake-up time of the test system within a paused session (test system wakes up the SUT)

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

PICS_CMN_CMN_Renegotiation	Indication for explicit testing of Renegotiation
PICS_CMN_CMN_Renegotiation LoopIndication	Indicates the number of iterations of the ChargingStatus (AC) or CurrentDemand (DC) loop, when the renegotiation is triggered by the test system (smaller or equal to PICS_CMN_CMN_LoopCounter). NOTE: The minimum value of loop counter is 30
PICS_CMN_CMN_LoopCounter	Indicates the number of iterations of the ChargingStatus (AC) or CurrentDemand (DC) loop NOTE: The minimum value of loop counter is 30

Table 7 — PICS for test system configurations CF_04_001 (SUT equals SECC)

PICS	Description
PICS_SECC_CMN_Pause	Indication for enabling a charging pause in SessionStopReq message (ChargingSession)
PICS_SECC_CMN_EIMDone	Indication for the specified state initiating an EIM Authorization Choice: i) beforePlugin, ii) afterPlugin, iii) duringSlac, iv) v2gAuthorization
PICS_SECC_CMN_EvStateInChargeProgress	Indication for explicit testing of State C or D of IEC 61851-1:2017, Annex A during charging
PICS_SECC_DC_WeldingDetection	Indication for explicit testing of WeldingDetection
PICS_SECC_CMN_DepartureTime	Indication for using a departure time in ChargeParameterDiscoveryReq message
PICS_SECC_CMN_MaxEntriesSAScheduleTuple	Indication for enabling a maximal number of expected entries in the SAScheduleTuple (received in ChargeParameterDiscoveryRes message) by using the element MaxEntriesSAScheduleTuple in ChargeParameterDiscoveryReq message

Table 8 — PICS for test system configurations CF_04_002 (SUT equals EVCC)

PICS	Description
PICS_EVCC_CMN_PmaxScheduleWithZeroPow	Indication for enabling a PmaxScheduleList with a list element pMax = 0W in SAScheduleList in ChargeParameterDiscoveryRes message NOTE: For this approach the following assumption is considered: The EV triggers a paused session if the EVSE cannot provide

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

	power for a specified time period. This approach might NOT work for all EVs. In this case all relevant test cases with reference to `PICS_EVCC_CMN_PmaxSchedulewithZeroPow` are out of the scope of the test execution. Choice: i) sleepWithoutCharge (0W in first element of PmaxScheduleList), ii) sleepAfterCharge (0W in second element of PmaxScheduleList), iii) none_
`PICS_EVCC_CMN_Receipt Required`	Indication for explicit testing of MeteringReceipt message activated by the element ReceiptRequired in ChargingStatusRes (AC) or CurrentDemandRes (DC) message
`PICS_CMN_EVCC_SalesTariff`	Indication for using a SalesTariff in ChargeParameterDiscoveryRes message

7.3.5 Protocol Implementation extra Information for Testing (PIXIT) Definition

Any additional information required for testing that is not covered in the form of an ICS / PICS, is specified in the form of Implementation extra Information for Testing (IXIT). In the context of a communication protocol as specified in ISO 15118-2 such statements are referred to as Protocol Implementation extra Information for Testing (PIXIT) cf. [ITU-T X.290] and are used to indicate to the SUT that a specific capability of the SUT is tested. In order to ensure testability of each message pair specified in ISO 15118-2, the following set of PIXIT are specified.

NOTE Due to the black box test paradigm in this document, it is not specified how to ensure that a corresponding PIXIT is set on the SUT side for a given executable test case.

All PIXIT specified in the ATS are summarized in Table 9 - Table 11. These PIXITs relate to the SUT and specify what protocol options are implemented in the SUT and what confiugration is required for each testcase

Table 9 — Selected PIXIT for test system configurations CF_04_001 and CF_04_002 (SUT either SECC or EVCC)

PIXIT	Description
`PIXIT_CMN_CMN_WakeUp`	Indication for the wake-up time of the SUT within a paused session (SUT wakes up the test system)
`PIXIT_CMN_CMN_PkiName`	Indication for selected PKI Choice: i) PKI_1, ii) PKI_2, iii) PKI_3, iv) PKI_4, v) PKI_5, vi) PKI_6, vii) PKI_7
`PIXIT_CMN_CMN_Renegotiation`	Indication for explicit testing of Renegotiation initiated by the SUT Choice: i) renegotiation, ii) unknown
`PIXIT_CMN_CMN_MeterReading`	Indication for explicit testing of MeterReading element Choice: i) meterReading, ii) unknown

Table 10 — Selected PIXIT for test system configurations CF_04_001 (SUT equals SECC)

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

PIXIT	Description
PIXIT_SECC_CMN_VAS	Indication for explicit testing of ServiceDetail message (SUT is supposed to send a serviceList within ServiceDiscoveryRes message) Choice: i) serviceDetail, ii) none_
PIXIT_SECC_CMN_ReceiptRequired	Indication for explicit testing of MeteringReceipt message
PIXIT_SECC_CMN_CertificateHandling	Indication for explicit testing of CertificateInstallation or CertificateUpdate message Choice: i) certInstall, ii) certUpdate, iii) none_
PIXIT_SECC_CMN_SalesTariff	Indication for explicit testing of Sales Tariff elements in ChargeParameterDiscoveryRes message Choice: i) salesTariff, ii) unknown
PIXIT_SECC_CMN_TLS	Indication for explicit testing of TLS in EIM mode
PIXIT_SECC_CMN_VersionNumberMinor	Indication for the supported VersionNumberMinor in SupportedAppProtocolRes message
PIXIT_SECC_CMN_InternetAccess	Indication for support of InternetAccess service in ServiceDiscoveryRes message Choice: i) internetAccess, ii) unknown
PIXIT_SECC_DC_CableCheckOngoing	Indication for explicit testing of the repetition of the CableCheck message sequence Choice: i) ongoing, ii) unknown
PIXIT_SECC_CMN_AuthorizationOngoing	Indication for explicit testing of the repetition of the Authorization message sequence (SUT is not authorized in case of EIM) Choice: i) ongoing, ii) unknown
PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing	Indication for explicit testing of the repetition of the ChargeParameterDiscovery message sequence Choice: i) ongoing, ii) unknown
PIXIT_SECC_CMN_MeterInfo	Indication for explicit testing of MeterInfo element in ChargingStatusRes (AC) or CurrentDemandRes (DC) message Choice: i) meterInfo, ii) unknown
PIXIT_SECC_CMN_PaymentOption	Indication for supported payment option within ServiceDiscoveryRes message (Only one payment option possible) Choice: i) externalPayment, ii) contract

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table 11 — Selected PIXIT for test system configurations CF_04_002 (SUT equals EVCC)

PIXIT	Description
PIXIT_EVCC_CMN_VAS	Indication for explicit testing of ServiceDetail message (SUT sends a ServiceDetailReq message for each service of the received serviceList) Choice: i) serviceDetail, ii) none_, iii) unknown
PIXIT_EVCC_CMN_Pause	Indication for enabling a charging pause in SessionStopReq message (ChargingSession) Choice: i) pause, ii) unknown, iii) none_
PIXIT_EVCC_DC_WeldingDetection	Indication for explicit testing of WeldingDetection message Choice: i) weldingDetection, ii) none_, iii) unknown
PIXIT_EVCC_CMN_TLS	Indication for explicit testing of TLS in EIM mode Choice: i) tls, ii) none_, iii) unknown
PIXIT_EVCC_CMN_CertificateHandling	Indication for explicit testing of CertificateInstallation or CertificateUpdate message Choice: i) certInstall, ii) certUpdate, iii) none_, iv) unknown NOTE: For an automated test execution of multiple certificate installation or certification update processes, a customer interaction in the SUT (e.g. Deletion of installed contract certificates) might be necessary.
PIXIT_EVCC_CMN_DepartureTime	Indication for explicit testing of DepartureTime element in ChargeParameterDiscoveryReq message Choice: i) departureTime, ii) none_, iii) unknown
PIXIT_EVCC_AC_MaxNumberOfPhasesInUse	Indication for explicit testing of MaxNumberOfPhasesInUse element in ChargeParameterDiscoveryReq message Choice: i) maxNumberOfPhasesInUse, ii) unknown
PIXIT_EVCC_CMN_ChargingProfileOptimization	Indication for the SUTs capability to optimize charging profiles based on the received SAScheduleList Choice: i) optimization, ii) none_, iii) unknown
PIXIT_EVCC_CMN_OldSAScheduleTupleID	Indication for explicit testing of using the previous SAScheduleTupleID after renegotiation in PowerDeliveryReq message Choice: i) oldSAScheduleTupleID, ii) unknown

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

PIXIT_EVCC_CMN_SAScheduleTupleIDMR	Indication for explicit testing SAScheduleTupleID within MeteringReceiptReq message Choice: i) sAScheduleTupleID, ii) unknown

7.4 Test suite identifiers

7.4.1 General information

The selection of common naming conventions is one simple and often used mechanism to implement test suites which are consistent, maintainable and understandable for multiple users. Therefore, based on common ETSI naming conventions more specific naming conventions for the ISO 15118 conformance test suite are specified.

7.4.2 Module identifiers

All modules specified in the TSS start with a capital letter. The ISO 15118 test suite specific module identifier for template, function or test case modules is specified as:

```
<modtype>_<sut>_<ctx>
```

The segments of this identifier are specified in Table 12. An example for test case module identifier is:

```
TestCases_SECC_SupportedAppProtocol
```

Table 12 — ISO 15118 test suite naming convention for modules

Identifier	Values	Description
<modtype>		Module type
	TestCases	Module including test cases
	Functions	Module including functions
	Templates	Module including templates
<sut>		System under test
	EVCC	Electric Vehicle Communication Controller
	SECC	Supply Equipment Communication Controller
	CMN	Common (exclusively for template modules)
<ctx>	{fullname}	Context (e.g. name of message pattern according to ISO 15118-2)

NOTE 1 to entry For module types other than templates, functions or test cases there is no identifier format specified.

7.4.3 Test case identifiers

The naming conventions for test cases are using a prefix, which is specified by ETSI as shown in Table 13.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table 13 — ETSI naming convention for test case names

Keyword	Definition	Example
testcase name	Every testcase begins with TC (`TC_`) `TC_TestCaseName`	`TC_SupportedAppProtocol`

The ISO 15118 test suite specific test case identifier is specified as:

`TC_<sut>_<dom>_<ttyp>_<ctx>_<nn>`

The segments of this identifier are described in Table 14. An example for test case identifier is:

`TC_SECC_AC_VTB_SupportedAppProtocol_001`

Table 14 — ISO 15118 test suite naming convention for test case identifiers

Identifier	Values	Description
`<prefix>`	`TC`	see Table 13
`<sut>`		System under test
	`EVCC`	Electric Vehicle Communication Controller
	`SECC`	Supply Equipment Communication Controller
`<dom>`		Domain
	`AC`	AC specific behaviors
	`DC`	DC specific behaviors
	`CMN`	Common
`<ttyp>`		Type of testing
	`VTB`	Valid test behavior
`<ctx>`	`{fullname}`	Context (e.g. name of message pattern according to ISO 15118-2)
`<nn>`	`{xxx}`	Sequential number from 001 to 999

7.4.4 Template identifiers

The naming conventions for templates are using a prefix, which is specified by ETSI as shown in Table 15.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table 15 — ETSI naming convention for templates

Keyword	Context	Definition	Example
template name	Templates with concrete attribute values	Every template begins with keyword m (`m_`)	`m_DNSRequest`
template name	Templates with wildcards or matching expression	If a template contains or refers to templates with wildcards {*|?} then template name begins with keyword mw (`mw_`)	`mw_DNSResponse`
template name	Templates with parameters, which do not assign or refer to templates with wildcards or matching expression	If a template contains attributes which are specified by parameters or constant values, then template name begins with keyword md (`md_`)	`md_DNSResponse` (integer ip)
template name	Templates with parameters, which do assign or refer to templates with wildcards or matching expression	If a template contains attributes which are specified by parameters, constant values or wildcards, then template name begins with keyword mdw (`mdw_`)	`mdw_DNSResponse` (integer ip)

The ISO 15118 test suite specific template identifier is specified as:

`<prefix>_<sut>_<dom>_<dtyp>_<nn>`

The segments of this identifier are described in Table 16. An example for template identifier is:

`mdw_EVCC_CMN_SupportedAppProtocol_001`

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table 16 — ISO 15118 test suite naming convention for template identifiers

Identifier	Values	Description
`<prefix>`		Type of template (see Table 15)
`<sut>`		System under test
	`EVCC`	Electric Vehicle Communication Controller
	`SECC`	Supply Equipment Communication Controller
	`CMN`	Common
`<dom>`		Domain
	`AC`	AC specific behaviors
	`DC`	DC specific behaviors
	`CMN`	Common
`<dtyp>`	`{fullname}`	Label of (root) data type according to ISO 15118-2
`<nn>`	`{xxx}`	Sequential number from 001 to 999

7.4.5 Function identifiers

The naming conventions for functions are using a prefix, which is specified by ETSI as shown in Table 17.

Table 17 — ETSI naming convention for function names

Keyword	Context	Definition	Example
function name	All functions	Every function begins with f (`f_`)	`f_functionName`

The ISO 15118 test suite naming convention for test case functions is specified as:

`<prefix>_<sut>_<dom>_<ftyp>_<ttyp>_<ctx>_<nn>`

The segments of this identifier are described in Table 18. An example for template identifier is:

`f_EVCC_AC_TB_VTB_SessionSetup_001`

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table 18 — ISO 15118 test suite naming convention for function names

Identifier	Values	Description
`<prefix>`		f_ (see Table 17)
`<sut>`		System under test
	`EVCC`	Electric vehicle communication controller
	`SECC`	Supply equipment communication controller
`<dom>`		Domain
	`AC`	AC specific behaviors
	`DC`	DC specific behaviors
	`CMN`	Common
`<ftyp>`		Function type
	`PR`	Pre-condition
	`TB`	Test behavior
	`PO`	Post-condition
`<ttyp>`		Type of testing
	`VTB`	Valid test behavior
`<ctx>`	`{fullname}`	Context (e.g. name of message pattern according to ISO 15118-2)
`<nn>`	`{xxx}`	Sequential number from 001 to 999

7.4.6 Timer identifiers

The naming conventions for timers are using a prefix, which is specified by ETSI as shown in Table 19.

Table 19 — ETSI naming convention for timers

Keyword	Context	Definition	Example
timer name	`t_`	Local timer	`t_wait`
timer name	`tc_`	Timer specified within a component	`tc_authMin`

The ISO 15118 test suite specific timer identifier is specified as:

`<prefix>_<ctx>`

The segments of this identifier are described in Table 20. An example for timer identifier is:

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

`tc_V2G_EVCC_Msg_Timer`

Table 20 — ISO 15118 test suite naming convention for timer identifiers

Identifier	Values	Description
`<prefix>`		Type of timer (see Table 19)
`<ctx>`	`{fullname}`	Context (for example, name of timer according to ISO 15118-2 or if not specified any given name describing the context of the timer, e.g. `EVCC_SDP_Timer`)

7.4.7 PICS/PIXIT identifiers

The ISO 15118 test suite naming convention for PICS/PIXIT is specified as:

`<pic>_<sut>_<dom>_<ctx>`

The segments of this identifier are described in Table 21. An example for PICS/PIXIT identifier is:

`PICS_CMN_CMN_IdentificationMode`

Table 21 — ISO 15118 test suite naming convention for PICS/PIXIT identifiers

Identifier	Values	Description
`<pic>`		Protocol implementation capability
	`PICS`	Protocol implementation conformance statement
	`PIXIT`	Protocol implementation extra information for testing
`<sut>`		System under test
	`EVCC`	Electric vehicle communication controller
	`SECC`	Supply equipment communication controller
	`CMN`	Common
`<dom>`		Domain
	`AC`	AC specific behaviors
	`DC`	DC specific behaviors
	`CMN`	Common
`<ctx>`	`{fullname}`	Context (e.g. name of message pattern according to ISO 15118-2)

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

7.4.8 Verdict identifiers

In this sub clause the conventions for test verdicts are specified. The test verdicts specified in this document are listed in Table 22.

Table 22 — ISO 15118 test suite conventions on verdict handling

Verdict type	TTCN-3 Definition	ISO 15118 test suite
none	Is implicitly assigned in the beginning of every test case by default and is reported as a final verdict in the absence of any other verdict assignment during the test case execution.	No TSS specific definition (see TTCN-3 definition).
pass	Means that everything is OK. A verdict given when the observed outcome satisfies the test purpose and is valid with respect to the relevant requirements and with respect to the PICS. [ITU-T X.290]	If in review of a requirement the SUT has a correct behavior, then this verdict type is used.
inconc	A verdict given when the observed outcome is valid with respect to the relevant requirements but prevents the test purpose from being accomplished. [ITU-T X.290]	Means that neither pass nor fail can be reliably assigned.
fail	A verdict given when the observed outcome is syntactically invalid or inopportune with respect to the relevant requirements or the PICS/PIXIT. [ITU-T X.290]	If in review of a requirement the SUT has a wrong behavior, then this verdict type is used.

7.5 Test suite coverage

The following conditions apply in terms of test case coverage for this document regarding requirements specified in ISO 15118-2.

— Requirements that are out of the scope of the ATS according to the conventions specified in this document or cannot be tested in black-box tests are summarized in Table 23.

— Requirements that are not explicitly tested through a dedicated test case in the ATS but implicitly tested by the SUT Adapter/Codec are summarized in Table 24.

Table 23 — Requirements of ISO 15118-2 not considered in the ATS

List of requirement IDs	Comment (no consideration in ATS)
[V2G2-133], [V2G2-162], [V2G2-439], [V2G2-449], [V2G2-663], [V2G2-678], [V2G2-698], [V2G2-703], [V2G2-707], [V2G2-709], [V2G2-719], [V2G2-720], [V2G2-734], [V2G2-735],	Cannot be tested explicitly in EVCC black box test configuration

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

List of requirement IDs	Comment (no consideration in ATS)
[V2G2-763], [V2G2-784], [V2G2-801], [V2G2-823], [V2G2-870], [V2G2-874], [V2G2-880], [V2G2-882], [V2G2-886], [V2G2-892], [V2G2-897], [V2G2-907], [V2G2-908], [V2G2-910], [V2G2-927], [V2G2-928]	
[V2G2-020], [V2G2-023], [V2G2-076], [V2G2-305], [V2G2-306], [V2G2-446], [V2G2-447], [V2G2-448]	Cannot be tested explicitly in EVCC black box test configuration due to strong configuration dependency on the SUT.
[V2G2-332], [V2G2-335], [V2G2-615], [V2G2-616]	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Out of scope of this standard (PIXIT consideration leads to strong increase of TC quantity) for the profile defined in the respective column.
[V2G2-134], [V2G2-146], [V2G2-219], [V2G2-233], [V2G2-234], [V2G2-301], [V2G2-317], [V2G2-322], [V2G2-325], [V2G2-328], [V2G2-336], [V2G2-337], [V2G2-338], [V2G2-444], [V2G2-540], [V2G2-542], [V2G2-544], [V2G2-546], [V2G2-547], [V2G2-549], [V2G2-550], [V2G2-552], [V2G2-553], [V2G2-555], [V2G2-556], [V2G2-558], [V2G2-559], [V2G2-561], [V2G2-562], [V2G2-564], [V2G2-566], [V2G2-567], [V2G2-569], [V2G2-570], [V2G2-572], [V2G2-574], [V2G2-578], [V2G2-579], [V2G2-581], [V2G2-583], [V2G2-585], [V2G2-586], [V2G2-588], [V2G2-589], [V2G2-592], [V2G2-595], [V2G2-596], [V2G2-598], [V2G2-629], [V2G2-639], [V2G2-668], [V2G2-690], [V2G2-726], [V2G2-727], [V2G2-774], [V2G2-796], [V2G2-798], [V2G2-803], [V2G2-808], [V2G2-824], [V2G2-826], [V2G2-853], [V2G2-857], [V2G2-872], [V2G2-877], [V2G2-883], [V2G2-887], [V2G2-902], [V2G2-916], [V2G2-923]	Cannot be tested explicitly in SECC black box test configuration
[V2G2-074], [V2G2-430], [V2G2-722], [V2G2-785], [V2G2-786], [V2G2-805]	Cannot be tested explicitly in SECC black box test configuration due to strong configuration dependency on the SUT.
[V2G2-011], [V2G2-012], [V2G2-089], [V2G2-090], [V2G2-092], [V2G2-094],	Cannot be tested explicitly in SECC/EVCC black box test configuration

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

List of requirement IDs	Comment (no consideration in ATS)
[V2G2-096], [V2G2-135], [V2G2-640], [V2G2-760], [V2G2-830], [V2G2-835], [V2G2-868], [V2G2-869], [V2G2-878], [V2G2-911], [V2G2-925]	
[V2G2-867]	Cannot be tested explicitly in SECC/EVCC black box test configuration .
[V2G2-280], [V2G2-281]	Cannot be tested explicitly in SECC/EVCC black box test configuration due to strong configuration dependency on the SUT.
[V2G2-318], [V2G2-324], [V2G2-333], [V2G2-334], [V2G2-339], [V2G2-340], [V2G2-749], [V2G2-755], [V2G2-772], [V2G2-775], [V2G2-802], [V2G2-806], [V2G2-807]	Out of scope of this standard (PIXIT consideration leads to strong increase of TC quantity)
[V2G2-001], [V2G2-002], [V2G2-003], [V2G2-014], [V2G2-016], [V2G2-017], [V2G2-037], [V2G2-038], [V2G2-039], [V2G2-040], [V2G2-041], [V2G2-042], [V2G2-043], [V2G2-044], [V2G2-045], [V2G2-046], [V2G2-047], [V2G2-049], [V2G2-050], [V2G2-051], [V2G2-052], [V2G2-053], [V2G2-054], [V2G2-055], [V2G2-057], [V2G2-058], [V2G2-059], [V2G2-060], [V2G2-061], [V2G2-062], [V2G2-063], [V2G2-064], [V2G2-065], [V2G2-067], [V2G2-070], [V2G2-071], [V2G2-163], [V2G2-164], [V2G2-638], [V2G2-649], [V2G2-650], [V2G2-693], [V2G2-694], [V2G2-695], [V2G2-810], [V2G2-811], [V2G2-843], [V2G2-844], [V2G2-850], [V2G2-873], [V2G2-876], [V2G2-885], [V2G2-924]	Out of scope of this standard (Requirement refers to external RFCs, standards, etc.)
[V2G2-289], [V2G2-290], [V2G2-291], [V2G2-292], [V2G2-314], [V2G2-315], [V2G2-329], [V2G2-330], [V2G2-480], [V2G2-643], [V2G2-644], [V2G2-674], [V2G2-676], [V2G2-743], [V2G2-761], [V2G2-777], [V2G2-833], [V2G2-834], [V2G2-837], [V2G2-838], [V2G2-851], [V2G2-858], [V2G2-859], [V2G2-860],	Out of scope of this standard (Requirement refers to power flow)

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

List of requirement IDs	Comment (no consideration in ATS)
[V2G2-862], [V2G2-863], [V2G2-865], [V2G2-866]	
[V2G2-035], [V2G2-036]	Out of scope of this standard (Requirements refers to ISO 15118-5)

Table 24 — Requirements of ISO 15118-2 implicitly covered by the SUT adapters / codecs

List of requirement IDs	Comment (implicitly tested)
[V2G2-010]	Cannot be tested explicitly in SECC/EVCC black box test configuration for the profile defined in the respective column. Tested implictly by V2G codec for the profile defined in the respective column
[V2G2-073], [V2G2-075], [V2G2-077], [V2G2-078], [V2G2-079], [V2G2-080], [V2G2-081], [V2G2-083], [V2G2-084], [V2G2-085], [V2G2-124], [V2G2-125], [V2G2-128], [V2G2-129], [V2G2-131], [V2G2-132], [V2G2-136], [V2G2-137], [V2G2-138], [V2G2-139], [V2G2-143], [V2G2-149], [V2G2-151], [V2G2-155], [V2G2-156], [V2G2-174], [V2G2-622], [V2G2-809], [V2G2-816]	Tested implictly by 15118-2 SUT adapter
[V2G2-097], [V2G2-098], [V2G2-099], [V2G2-100], [V2G2-101], [V2G2-102], [V2G2-171], [V2G2-440], [V2G2-445], [V2G2-600], [V2G2-697], [V2G2-825]	Tested implictly by V2G codec
[V2G2-651]	Tested implictly by V2G codec. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column.

The resulting test suite coverage in this document with reference to requirements in ISO 15118-2 is summarized in Table 25. It only includes requirements which are not listed for exclusion in Table 23 or Table 24. Table 25 specifies the relevance of requirements of ISO 15118-2 for the ATS, based upon the type of SUT (EVCC or SECC), the authentication profile (EIM or PnC) and AC or DC charging. The following symbols are used in Table 25:

X Indicates requirements that are covered in the ATS with one or more test cases.

I Indicates requirements that are indirectly covered in the ATS but other than those listed in Table 24 can be directly associated to one or more test cases in the ATS.

P Indicates requirements that are only partially covered by one or more test cases in the ATS.

N Requirements that are not testable for the profile in the respective column.

- Requirements that are not applicable for the profile in the respective column.

O Requirements that are out of the scope of the ATS for the profile in the respective column.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table 25 — ATS coverage of requirements in ISO 15118-2

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-004]	N	N	X	X	N	N	X	X	Cannot be tested explicitly in SECC/EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-005]	-	-	P	P	-	-	X	X	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially for the profile defined in the respective column Group_[V2G2-005]_[V2G2-007]
[V2G2-006]	-	-	-	-	-	-	X	X	Group_[V2G2-006]_[V2G2-652]_[V2G2-909]
[V2G2-007]	-	-	P	P	-	-	X	X	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially for the profile defined in the respective column Group_[V2G2-005]_[V2G2-007]
[V2G2-008]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-166]. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column TC_EVCC_CMN_VTB_SupportedAppProtocol_001
[V2G2-009]	N	N	X	X	N	N	X	X	Cannot be tested explicitly in SECC/EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-018]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-142] Group_[V2G2-142]_[V2G2-623]_[V2G2-018]_[V2G2-019]
[V2G2-019]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-142] Group_[V2G2-142]_[V2G2-623]_[V2G2-018]_[V2G2-019]
[V2G2-021]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column TC_EVCC_CMN_VTB_SupportedAppProtocol_001
[V2G2-022]	-	-	-	-	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-024]	-	-	-	-	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-025]	-	-	-	-	X	X	X	X	Group_[V2G2-728]_[V2G2-025]
[V2G2-026]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-168] TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_006 TC_SECC_CMN_VTB_SupportedAppProtocol_004
[V2G2-027]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-027] TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_007 TC_SECC_CMN_VTB_SDP_003

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-029]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-723] TC_SECC_CMN_VTB_SDP_004 TC_SECC_CMN_VTB_SDP_005 TC_SECC_CMN_VTB_SDP_006
[V2G2-030]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-168] TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_006 TC_SECC_CMN_VTB_SupportedAppProtocol_004
[V2G2-031]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-168] TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_006 TC_SECC_CMN_VTB_SupportedAppProtocol_004
[V2G2-032]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-032] TC_SECC_CMN_VTB_SupportedAppProtocol_005
[V2G2-033]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-168] TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_006 TC_SECC_CMN_VTB_SupportedAppProtocol_004
[V2G2-034]	X	X	X	X	-	-	-	-	Group_[V2G2-729]_[V2G2-034]
[V2G2-068]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-168]. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_006 TC_SECC_CMN_VTB_SupportedAppProtocol_004
[V2G2-082]	X	X	X	X	X	X	X	X	Group_[V2G2-082]_[V2G2-086]
[V2G2-086]	X	X	X	X	X	X	X	X	Group_[V2G2-082]_[V2G2-086]
[V2G2-087]	P	P	P	P	P	P	P	P	Table 9 and Table 10 can only be tested partially TC_EVCC_CMN_VTB_V2GTPSDP_001 TC_EVCC_CMN_VTB_V2GTPSessionSetup_001 TC_SECC_CMN_VTB_V2GTPSDP_001 TC_SECC_CMN_VTB_V2GTPSessionSetup_001
[V2G2-088]	P	P	P	P	P	P	P	P	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases TC_EVCC_CMN_VTB_V2GTPSessionSetup_001 TC_SECC_CMN_VTB_V2GTPSessionSetup_001
[V2G2-103]	-	-	X	X	-	-	X	X	Group_[V2G2-103]_[V2G2-104]
[V2G2-104]	-	-	P	P	-	-	P	P	Internal states cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-103]_[V2G2-104]
[V2G2-108]	-	-	-	-	-	-	X	X	TC_EVCC_CMN_VTB_PaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_002

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-117]	-	-	-	-	-	-	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-119]	-	-	-	-	-	-	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-121]	-	-	-	-	-	-	I	I	Tested indirectly through test for requirement [V2G2-500]/[V2G2-501] and [V2G2-503] TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-122]	-	-	-	-	-	-	I	I	Tested indirectly through test for requirement [V2G2-500]/[V2G2-501] and [V2G2-503] TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-123]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_007 TC_SECC_CMN_VTB_SDP_003
[V2G2-126]	-	-	-	-	X	X	X	X	TC_EVCC_CMN_VTB_SDP_006
[V2G2-127]	-	-	-	-	P	P	P	P	Subclause 7.8.3.1 can only be tested partially TC_EVCC_CMN_VTB_V2GTPSDP_001
[V2G2-130]	P	P	P	P	-	-	-	-	Subclause 7.8.3.1 can only be tested partially TC_SECC_CMN_VTB_V2GTPSDP_001
[V2G2-140]	-	-	-	-	X	X	X	X	TC_EVCC_CMN_VTB_V2GTPSDP_001
[V2G2-141]	-	-	-	-	X	X	X	X	TC_EVCC_CMN_VTB_V2GTPSDP_001
[V2G2-142]	-	-	-	-	X	X	X	X	Group_[V2G2-142]_[V2G2-623]_[V2G2-018]_[V2G2-019]
[V2G2-144]	P	P	P	P	-	-	-	-	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_007 TC_SECC_CMN_VTB_SDP_003
[V2G2-145]	-	-	-	-	P	P	P	P	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases TC_EVCC_CMN_VTB_SDP_007
[V2G2-147]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_007 TC_SECC_CMN_VTB_SDP_003
[V2G2-150]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_007 TC_SECC_CMN_VTB_SDP_003
[V2G2-152]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_V2GTPSDP_001
[V2G2-153]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_V2GTPSDP_001

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-154]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_007 TC_SECC_CMN_VTB_SDP_003
[V2G2-157]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-161] Group_[V2G2-157]_[V2G2-160]_[V2G2-161]
[V2G2-158]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-157] and an initiated paused session. Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-158]_[V2G2-731]
[V2G2-159]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-160] Group_[V2G2-159]
[V2G2-160]	-	-	-	-	X	X	X	X	Group_[V2G2-157]_[V2G2-160]_[V2G2-161]
[V2G2-161]	-	-	-	-	I	I	I	I	Group_[V2G2-157]_[V2G2-160]_[V2G2-161] Group_[V2G2-161]
[V2G2-165]	X	X	X	X	X	X	X	X	Group_[V2G2-165]_[V2G2-178]
[V2G2-166]	-	-	-	-	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-167]	-	-	-	-	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-168]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_006 TC_SECC_CMN_VTB_SupportedAppProtocol_004
[V2G2-169]	P	P	P	P	-	-	-	-	Priority handling cannot be tested explicitly in SECC black box test configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_006 TC_SECC_CMN_VTB_SupportedAppProtocol_004
[V2G2-170]	X	X	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-172]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_SupportedAppProtocol_002
[V2G2-173]	-	-	-	-	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_002
[V2G2-175]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-176]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_006 TC_SECC_CMN_VTB_SupportedAppProtocol_004
[V2G2-178]	P	P	P	P	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									Group_[V2G2-165]_[V2G2-178]
[V2G2-179]	X	X	X	X	X	X	X	X	Group_[V2G2-179]_[V2G2-180]_[V2G2-181]_[V2G2-183]
[V2G2-180]	X	X	X	X	X	X	X	X	Group_[V2G2-179]_[V2G2-180]_[V2G2-181]_[V2G2-183]
[V2G2-181]	X	X	X	X	X	X	X	X	Group_[V2G2-179]_[V2G2-180]_[V2G2-181]_[V2G2-183]
[V2G2-182]	X	X	X	X	X	X	X	X	Group_[V2G2-182]
[V2G2-183]	X	X	X	X	X	X	X	X	Group_[V2G2-179]_[V2G2-180]_[V2G2-181]_[V2G2-183]
[V2G2-188]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-188]_[V2G2-189]_[V2G2-485]
[V2G2-189]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-188]_[V2G2-189]_[V2G2-485]
[V2G2-190]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-190]_[V2G2-191]_[V2G2-543]_[V2G2-192]
[V2G2-191]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-190]_[V2G2-191]_[V2G2-543]_[V2G2-192]
[V2G2-192]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-190]_[V2G2-191]_[V2G2-543]_[V2G2-192]
[V2G2-193]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_ServiceDiscovery_001
[V2G2-194]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_ServiceDiscovery_001
[V2G2-195]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-195]_[V2G2-196]
[V2G2-196]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-195]_[V2G2-196]
[V2G2-197]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_ServiceDetail_002 TC_EVCC_CMN_VTB_ServiceDetail_001
[V2G2-198]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_ServiceDetail_002 TC_EVCC_CMN_VTB_ServiceDetail_001
[V2G2-199]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-199]_[V2G2-200]

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-200]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-199]_[V2G2-200]
[V2G2-201]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-202]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-203]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-203]_[V2G2-204]
[V2G2-204]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-203]_[V2G2-204]
[V2G2-205]	-	-	N	N	-	-	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_002 TC_EVCC_CMN_VTB_PaymentDetails_003
[V2G2-206]	-	-	N	N	-	-	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_002 TC_EVCC_CMN_VTB_PaymentDetails_003
[V2G2-208]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-208]_[V2G2-209]
[V2G2-209]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-208]_[V2G2-209]
[V2G2-210]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_Authorization_002 TC_EVCC_CMN_VTB_Authorization_001 TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-211]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_CMN_VTB_Authorization_002 TC_EVCC_CMN_VTB_Authorization_001 TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-212]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-212]_[V2G2-213]
[V2G2-213]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-212]_[V2G2-213]
[V2G2-214]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-214]_[V2G2-216]
[V2G2-215]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered. Group_[V2G2-215]
[V2G2-216]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-214]_[V2G2-216]
[V2G2-218]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-218]_[V2G2-220]
[V2G2-220]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-218]_[V2G2-220]
[V2G2-221]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-221]_[V2G2-222]_[V2G2-286] [V2G2-510]
[V2G2-222]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-221]_[V2G2-222]_[V2G2-286] [V2G2-510]
[V2G2-223]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-223]_[V2G2-226]
[V2G2-224]	X	X	X	X	-	-	-	-	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases TC_SECC_AC_VTB_PowerDelivery_001 TC_SECC_AC_VTB_PowerDelivery_002 TC_SECC_DC_VTB_PowerDelivery_001
[V2G2-225]	X	X	X	X	-	-	-	-	TC_SECC_AC_VTB_PowerDelivery_010 TC_SECC_DC_VTB_PowerDelivery_007
[V2G2-226]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-223]_[V2G2-226]
[V2G2-227]	-	-	-	-	-	-	X	X	TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-228]	-	-	N	N	-	-	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateUpdate_001

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-229]	-	-	N	N	-	-	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-230]	-	-	-	-	-	-	I	I	Tested indirectly through test for requirement [V2G2-501] and [V2G2-503] TC_EVCC_CMN_VTB_Authorization_011
[V2G2-231]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-231]_[V2G2-232]
[V2G2-232]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-231]_[V2G2-232]
[V2G2-235]	-	-	N	N	-	-	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-236]	-	-	N	N	-	-	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-237]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-237]_[V2G2-238]
[V2G2-238]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-237]_[V2G2-238]
[V2G2-239]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-239]_[V2G2-738]
[V2G2-240]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-240]_[V2G2-241]
[V2G2-241]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-240]_[V2G2-241]
[V2G2-242]	N	-	N	-	X	-	X	-	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-242]
[V2G2-243]	X	-	X	-	N	-	N	-	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-243]_[V2G2-244]
[V2G2-244]	X	-	X	-	N	-	N	-	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-243]_[V2G2-244]
[V2G2-245]	-	-	N	N	-	-	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-245]_[V2G2-246]

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-246]	-	-	N	N	-	-	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-245]_[V2G2-246]
[V2G2-247]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-247]_[V2G2-248]
[V2G2-248]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-247]_[V2G2-248]
[V2G2-249]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_DC_VTB_CableCheck_001
[V2G2-250]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column TC_EVCC_DC_VTB_CableCheck_001
[V2G2-251]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-251]_[V2G2-252]
[V2G2-252]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-251]_[V2G2-252]
[V2G2-253]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-253]_[V2G2-254]
[V2G2-254]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-253]_[V2G2-254]
[V2G2-255]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_DC_VTB_PreCharge_001 TC_SECC_DC_VTB_PreCharge_003
[V2G2-256]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_DC_VTB_PreCharge_001 TC_SECC_DC_VTB_PreCharge_003
[V2G2-257]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-257]_[V2G2-258]
[V2G2-258]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-257]_[V2G2-258]
[V2G2-259]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-259]_[V2G2-260]
[V2G2-260]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									Group_[V2G2-259]_[V2G2-260]
[V2G2-261]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_DC_VTB_WeldingDetection_001 TC_EVCC_DC_VTB_WeldingDetection_009
[V2G2-262]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_DC_VTB_WeldingDetection_001 TC_EVCC_DC_VTB_WeldingDetection_009
[V2G2-263]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_DC_VTB_WeldingDetection_001 TC_SECC_DC_VTB_WeldingDetection_003 TC_SECC_DC_VTB_WeldingDetection_005
[V2G2-264]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_DC_VTB_WeldingDetection_001 TC_SECC_DC_VTB_WeldingDetection_003 TC_SECC_DC_VTB_WeldingDetection_005
[V2G2-265]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-266]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-267]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-268]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-271]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-272]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-274]	-	-	X	X	-	-	X	X	TC_EVCC_CMN_VTB_CertificateUpdate_001 TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-275]	-	-	X	X	-	-	X	X	TC_EVCC_CMN_VTB_CertificateUpdate_001 TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-276]	X	X	X	X	-	-	X	X	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-276]_[V2G2-277]
[V2G2-277]	X	X	X	X	-	-	X	X	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-276]_[V2G2-277]
[V2G2-278]	X	X	X	X	X	X	X	X	Group_[V2G2-278]_[V2G2-279]
[V2G2-279]	X	X	X	X	X	X	X	X	Group_[V2G2-278]_[V2G2-279]
[V2G2-282]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-283]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-284]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]
[V2G2-286]	-	-	-	-	X	X	X	X	Group_[V2G2-221]_[V2G2-222]_[V2G2-286] [V2G2-510]
[V2G2-288]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-293]	-	-	-	-	X	X	X	X	Group_[V2G2-293]_[V2G2-673]
[V2G2-294]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]
[V2G2-296]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-296]
[V2G2-297]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-298]. Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-297]_[V2G2-298]
[V2G2-298]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-297]_[V2G2-298]
[V2G2-299]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]
[V2G2-300]	X	X	X	X	-	-	-	-	Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]
[V2G2-303]	P	P	P	P	-	-	-	-	SalesTariff is only testable if a corresponding PIXIT is considered Group_[V2G2-303]
[V2G2-304]	P	P	P	P	-	-	-	-	SalesTariff is only testable if a corresponding PIXIT is considered Group_[V2G2-304]
[V2G2-307]	-	-	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered Group_([V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_([V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-308]	-	-	I	I	-	-	-	-	Tested indirectly through test of signature validation. Requirement is only testable if a corresponding PIXIT is considered Group_([V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_([V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-309]	-	-	I	I	-	-	-	-	Tested indirectly through test of signature validation. Requirement is only testable if a corresponding PIXIT is considered Group_([V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_([V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-310]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-313]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-316]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									Group_([V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_([V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-321]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_([V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_([V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-327]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-331]	P	P	P	P	-	-	-	-	SalesTariff is only testable if a corresponding PIXIT is considered Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-343]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-344]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-345]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-346]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-347]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-348]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-349]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-350]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-351]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-352]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-353]	-	-	X	X	-	-	X	X	Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-354]	-	-	X	X	-	-	X	X	Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-355]	-	-	N	N	-	-	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateInstallation_001 TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-356]	-	-	N	N	-	-	X	X	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateInstallation_001 TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-358]	X	-	X	-	N	-	N	-	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-358]_[V2G2-359]
[V2G2-359]	X	-	X	-	N	-	N	-	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-358]_[V2G2-359]
[V2G2-360]	N	-	N	-	X	-	X	-	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-360]_[V2G2-361]

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-361]	N	-	N	-	X	-	X	-	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-360]_[V2G2-361]
[V2G2-362]	X	-	X	-	N	-	N	-	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-573]_[V2G2-362]_[V2G2-363]
[V2G2-363]	X	-	X	-	N	-	N	-	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-573]_[V2G2-362]_[V2G2-363]
[V2G2-364]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-364]_[V2G2-365]_[V2G2-366]
[V2G2-365]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-364]_[V2G2-365]_[V2G2-366]
[V2G2-366]	-	P	-	P	-	-	-	-	Table 98 can only be tested partially Group_[V2G2-364]_[V2G2-365]_[V2G2-366]
[V2G2-367]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-367]_[V2G2-368]_[V2G2-369]
[V2G2-368]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-367]_[V2G2-368]_[V2G2-369]
[V2G2-369]	-	-	-	-	-	P	-	P	Table 100 can only be tested partially Group_[V2G2-367]_[V2G2-368]_[V2G2-369]
[V2G2-370]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-370]_[V2G2-371]
[V2G2-371]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-370]_[V2G2-371]
[V2G2-372]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-372]_[V2G2-373]
[V2G2-373]	-	X	-	X	-	N	-	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-372]_[V2G2-373]
[V2G2-374]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-374]_[V2G2-375]
[V2G2-375]	-	N	-	N	-	X	-	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-374]_[V2G2-375]
[V2G2-376]	-	-	-	-	P	-	-	-	Table 104 can only be tested partially Group_[V2G2-376]
[V2G2-377]	P	-	-	-	-	-	-	-	Table 104 can only be tested partially Group_[V2G2-377]

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-378]	-	-	-	-	X	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_002
[V2G2-379]	X	-	-	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011
[V2G2-380]	-	-	-	-	-	-	P	-	Table 104 can only be tested partially Group_[V2G2-380]
[V2G2-381]	-	-	P	-	-	-	-	-	Table 104 can only be tested partially Group_[V2G2-381]
[V2G2-384]	-	-	-	-	-	-	X	-	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_001
[V2G2-385]	-	-	X	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-386]	-	-	-	-	-	-	X	-	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-387]	-	-	X	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_CertificateInstallation_001
[V2G2-388]	-	-	-	-	-	-	X	-	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-389]	-	-	X	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-390]	-	-	-	-	-	P	-	-	Table 104 can only be tested partially Group_[V2G2-390]
[V2G2-391]	-	P	-	-	-	-	-	-	Table 104 can only be tested partially Group_[V2G2-391]
[V2G2-392]	-	-	-	-	-	X	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_002
[V2G2-393]	-	X	-	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011
[V2G2-394]	-	-	-	-	-	-	-	P	Table 104 can only be tested partially Group_[V2G2-394]
[V2G2-395]	-	-	-	P	-	-	-	-	Table 104 can only be tested partially Group_[V2G2-395]
[V2G2-396]	-	-	-	-	-	-	-	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_001
[V2G2-397]	-	-	-	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-398]	-	-	-	-	-	-	-	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-399]	-	-	-	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_CertificateInstallation_001
[V2G2-400]	-	-	-	-	-	-	-	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateUpdate_001

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-401]	-	-	-	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-402]	-	-	-	-	P	-	-	-	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-402]
[V2G2-403]	-	-	-	-	-	P	-	-	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-403]
[V2G2-404]	-	-	-	-	-	-	P	-	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-404]
[V2G2-405]	-	-	-	-	-	-	-	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-405]
[V2G2-406]	-	-	X	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_AC_VTB_ChargingStatus_002 TC_SECC_AC_VTB_ChargingStatus_007 TC_SECC_AC_VTB_ChargingStatus_009
[V2G2-407]	-	-	X	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_AC_VTB_ChargingStatus_001 TC_SECC_AC_VTB_ChargingStatus_006 TC_SECC_AC_VTB_ChargingStatus_008
[V2G2-408]	-	-	-	-	-	-	X	-	Group_[V2G2-512]_[V2G2-408]
[V2G2-410]	-	-	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_CertificateInstallation_001
[V2G2-411]	-	-	-	-	-	-	X	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-412]	-	-	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-413]	-	-	-	-	-	-	X	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-414]	X	X	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-415]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_002 TC_EVCC_CMN_VTB_ServiceDetail_001
[V2G2-416]	P	P	P	P	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-417]	N	N	N	N	P	P	P	P	Cannot be tested explicitly in SECC black box test configuration. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_ServiceDetail_002 TC_EVCC_CMN_VTB_ServiceDetail_001
[V2G2-418]	X	X	X	X	-	-	-	-	TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-419]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirements [V2G2-420] and [V2G2-421] TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-420]	P	P	P	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-421]	P	P	P	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-422]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_002 TC_EVCC_CMN_VTB_ServiceDetail_001
[V2G2-424]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered TC_EVCC_CMN_VTB_ServiceDetail_002 TC_EVCC_CMN_VTB_ServiceDetail_001
[V2G2-425]	X	X	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_007 TC_SECC_CMN_VTB_ServiceDetail_008
[V2G2-426]	X	X	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-427]	X	X	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-428]	-	-	P	P	-	-	-	-	Table 106 can only be tested partially TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-429]	P	P	P	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Table 107 can only be tested partially

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-431]	-	-	-	-	X	X	X	X	Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]
[V2G2-432]	-	-	-	-	X	X	X	X	Group_[V2G2-493]_[V2G2-494]_[V2G2-432]
[V2G2-433]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_PaymentServiceSelection_006
[V2G2-434]	-	-	-	-	P	P	P	P	Table 108 and 109 can only be tested partially Group_[V2G2-434]_[V2G2-436]_[V2G2-437]_[V2G2-438]
[V2G2-435]	P	P	P	P	-	-	-	-	Table 108 and 109 can only be tested partially Group_[V2G2-435]
[V2G2-436]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-438] Group_[V2G2-434]_[V2G2-436]_[V2G2-437]_[V2G2-438]
[V2G2-437]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-438] Group_[V2G2-434]_[V2G2-436]_[V2G2-437]_[V2G2-438]
[V2G2-438]	-	-	-	-	P	P	P	P	Internal states cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-434]_[V2G2-436]_[V2G2-437]_[V2G2-438]
[V2G2-441]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-443] Group_[V2G2-441]_[V2G2-442]_[V2G2-443]
[V2G2-442]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-443] Group_[V2G2-441]_[V2G2-442]_[V2G2-443]
[V2G2-443]	P	P	P	P	-	-	-	-	Internal states cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-441]_[V2G2-442]_[V2G2-443]
[V2G2-457]	P	P	P	P	-	-	-	-	Only one message sequence will be considered, so the requirement can only be tested partially Group_[V2G2-457]
[V2G2-458]	X	X	X	X	-	-	-	-	Group_[V2G2-458]
[V2G2-459]	X	X	X	X	-	-	-	-	Only one message sequence will be considered, so the requirement can only be tested partially Group_[V2G2-459]_[V2G2-538]
[V2G2-460]	X	X	X	X	-	-	-	-	Only one message sequence will be considered, so the requirement can only be tested partially Group_[V2G2-460]
[V2G2-461]	-	-	X	X	-	-	-	-	Group_[V2G2-461]
[V2G2-462]	X	X	X	X	-	-	-	-	Group_[V2G2-462]
[V2G2-463]	X	X	X	X	-	-	-	-	TC_SECC_AC_VTB_SessionSetup_001 TC_SECC_AC_VTB_SessionSetup_004 TC_SECC_DC_VTB_SessionSetup_001 TC_SECC_DC_VTB_SessionSetup_004
[V2G2-464]	X	X	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_CMN_VTB_ServiceDetail_007 TC_SECC_CMN_VTB_ServiceDetail_008
[V2G2-465]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_PaymentServiceSelection_009 TC_SECC_CMN_VTB_PaymentServiceSelection_010
[V2G2-467]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_PaymentServiceSelection_006

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-468]	-	-	X	X	-	-	-	-	TC_SECC_CMN_VTB_CertificateInstallation_007
[V2G2-470]	-	-	X	X	-	-	-	-	TC_SECC_CMN_VTB_CertificateUpdate_008 TC_SECC_CMN_VTB_CertificateUpdate_009 TC_SECC_CMN_VTB_CertificateUpdate_010 TC_SECC_CMN_VTB_CertificateUpdate_011
[V2G2-473]	-	-	X	X	-	-	-	-	TC_SECC_CMN_VTB_CertificateUpdate_016
[V2G2-474]	-	-	X	X	-	-	-	-	TC_SECC_CMN_VTB_PaymentDetails_007
[V2G2-475]	-	-	X	X	-	-	-	-	TC_SECC_CMN_VTB_Authorization_007
[V2G2-476]	X	X	X	X	-	-	-	-	TC_SECC_AC_VTB_ChargeParameterDiscovery_004 TC_SECC_DC_VTB_ChargeParameterDiscovery_003
[V2G2-477]	X	X	X	X	-	-	-	-	TC_SECC_AC_VTB_ChargeParameterDiscovery_005 TC_SECC_DC_VTB_ChargeParameterDiscovery_004
[V2G2-478]	X	X	X	X	-	-	-	-	TC_SECC_AC_VTB_PowerDelivery_010 TC_SECC_DC_VTB_PowerDelivery_007
[V2G2-479]	X	X	X	X	-	-	-	-	TC_SECC_AC_VTB_PowerDelivery_011 TC_SECC_DC_VTB_PowerDelivery_008
[V2G2-481]	-	-	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-481]
[V2G2-482]	-	-	-	-	P	P	P	P	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases Group_[V2G2-482]
[V2G2-483]	-	-	-	-	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-484]	-	-	-	-	X	X	X	X	TC_EVCC_CMN_VTB_SupportedAppProtocol_003
[V2G2-485]	-	-	-	-	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-188]_[V2G2-189]_[V2G2-485]
[V2G2-486]	-	-	-	-	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_CMN_VTB_SessionSetup_002 TC_EVCC_CMN_VTB_SessionSetup_003 TC_EVCC_CMN_VTB_SessionSetup_004
[V2G2-487]	-	-	-	-	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_ServiceDiscovery_001
[V2G2-488]	-	-	-	-	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_CMN_VTB_ServiceDiscovery_002 TC_EVCC_CMN_VTB_ServiceDiscovery_003 TC_EVCC_CMN_VTB_ServiceDiscovery_004 TC_EVCC_CMN_VTB_ServiceDiscovery_005
[V2G2-489]	-	-	-	-	P	P	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_EVCC_CMN_VTB_ServiceDetail_002 TC_EVCC_CMN_VTB_ServiceDetail_001
[V2G2-490]	-	-	-	-	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-490]
[V2G2-491]	-	-	-	-	P	P	P	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-491]
[V2G2-492]	-	-	-	-	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-492]
[V2G2-493]	-	-	-	-	P	P	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-493]_[V2G2-494]_[V2G2-432]
[V2G2-494]	-	-	-	-	P	P	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-493]_[V2G2-494]_[V2G2-432]
[V2G2-495]	-	-	-	-	-	-	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_PaymentDetails_003
[V2G2-496]	-	-	-	-	-	-	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-497]	-	-	-	-	-	-	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-498]	-	-	-	-	-	-	P	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-498]
[V2G2-499]	-	-	-	-	-	-	P	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-499]
[V2G2-500]	-	-	-	-	-	-	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_PaymentDetails_001

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-501]	-	-	-	-	-	-	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_PaymentDetails_002
[V2G2-502]	-	-	-	-	-	-	P	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement can only be tested partially TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002 TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003 TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004 TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005
[V2G2-503]	-	-	-	-	-	-	P	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_Authorization_001 TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-504]	-	-	-	-	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-504]
[V2G2-505]	-	-	-	-	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-505]
[V2G2-506]	-	-	-	-	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-506]
[V2G2-507]	-	-	-	-	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-507]
[V2G2-508]	-	-	-	-	X	X	X	X	Group_[V2G2-508]
[V2G2-509]	-	-	-	-	P	P	-	-	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_Authorization_002
[V2G2-510]	-	-	-	-	P	-	P	-	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-510]_[V2G2-847]_[V2G2-846]
[V2G2-511]	-	-	-	-	P	-	P	-	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingStatus_002 TC_EVCC_AC_VTB_ChargingStatus_003 TC_EVCC_AC_VTB_ChargingStatus_004 TC_EVCC_AC_VTB_ChargingStatus_005
[V2G2-512]	-	-	-	-	-	-	P	-	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									Group_[V2G2-512]_[V2G2-408]
[V2G2-513]	-	-	-	-	P	-	P	-	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013
[V2G2-514]	-	-	-	-	P	-	P	-	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingStatus_001
[V2G2-516]	-	-	-	-	P	-	P	-	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013
[V2G2-517]	-	-	-	-	-	-	P	-	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-517]
[V2G2-518]	-	-	-	-	-	-	P	-	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014
[V2G2-519]	-	-	-	-	-	-	P	-	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014
[V2G2-520]	-	-	-	-	P	-	P	-	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-520]
[V2G2-521]	-	-	-	-	X	-	X	-	Requirement is only testable if a corresponding PIXIT is considered. TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013
[V2G2-522]	-	-	-	-	-	-	X	-	Requirement is only testable if a corresponding PIXIT is considered. TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014
[V2G2-524]	-	-	-	-	-	P	-	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_DC_VTB_CableCheck_002 TC_EVCC_DC_VTB_CableCheck_003 TC_EVCC_DC_VTB_CableCheck_004 TC_EVCC_DC_VTB_CableCheck_005
[V2G2-525]	-	-	-	-	-	P	-	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_PreCharge_001
[V2G2-526]	-	-	-	-	-	P	-	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_DC_VTB_PreCharge_002 TC_EVCC_DC_VTB_PreCharge_003

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_EVCC_DC_VTB_PreCharge_004 TC_EVCC_DC_VTB_PreCharge_005
[V2G2-527]	-	-	-	-	-	P	-	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001
[V2G2-528]	-	-	-	-	-	P	-	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-528]
[V2G2-530]	-	-	-	-	-	P	-	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_CurrentDemand_001
[V2G2-531]	-	-	-	-	-	P	-	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013
[V2G2-532]	-	-	-	-	-	P	-	P	Only one case can be considered, so the requirement can only be tested partially TC_EVCC_DC_VTB_CurrentDemand_002 TC_EVCC_DC_VTB_CurrentDemand_003 TC_EVCC_DC_VTB_CurrentDemand_004 TC_EVCC_DC_VTB_CurrentDemand_005
[V2G2-533]	-	-	-	-	-	P	-	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_WeldingDetection_001 TC_EVCC_DC_VTB_WeldingDetection_009
[V2G2-534]	-	-	-	-	-	P	-	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement can only be tested partially TC_EVCC_DC_VTB_WeldingDetection_002 TC_EVCC_DC_VTB_WeldingDetection_003 TC_EVCC_DC_VTB_WeldingDetection_004 TC_EVCC_DC_VTB_WeldingDetection_005
[V2G2-535]	-	-	-	-	-	P	-	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-535]
[V2G2-536]					-	-	-	-	Tested indirectly through test for requirement [V2G2-537]
[V2G2-537]					-	-	-	-	
[V2G2-538]	P	P	P	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-459]_[V2G2-538]
[V2G2-539]	X	X	X	X	-	-	-	-	Group_[V2G2-539]_[V2G2-736]
[V2G2-541]	P	P	P	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_SupportedAppProtocol_001

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_CMN_VTB_SupportedAppProtocol_006 TC_SECC_CMN_VTB_SupportedAppProtocol_004
[V2G2-543]	P	P	P	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-190]_[V2G2-191]_[V2G2-543]_[V2G2-192]
[V2G2-545]	P	P	P	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-548]	P	P	P	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_ServiceDetail_001 TC_SECC_CMN_VTB_ServiceDetail_011 TC_SECC_CMN_VTB_ServiceDetail_002 TC_SECC_CMN_VTB_ServiceDetail_012
[V2G2-551]	P	P	P	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_PaymentServiceSelection_001 TC_SECC_CMN_VTB_PaymentServiceSelection_002 TC_SECC_CMN_VTB_PaymentServiceSelection_003
[V2G2-554]	-	-	P	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_CertificateInstallation_001
[V2G2-557]	-	-	P	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_CertificateUpdate_001 TC_SECC_CMN_VTB_CertificateUpdate_002
[V2G2-560]	-	-	P	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_PaymentDetails_001 TC_SECC_CMN_VTB_PaymentDetails_002 TC_SECC_CMN_VTB_PaymentDetails_003
[V2G2-563]	P	P	P	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-563]_[V2G2-856]
[V2G2-568]	P	P	P	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_005 TC_SECC_DC_VTB_SessionStop_001
[V2G2-571]	P	P	P	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_AC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_002
[V2G2-573]	P	-	P	-	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-573]_[V2G2-362]_[V2G2-363]
[V2G2-575]	P	-	P	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-575]_[V2G2-577]
[V2G2-576]	P	-	P	-	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_001 TC_SECC_AC_VTB_PowerDelivery_002 TC_SECC_AC_VTB_PowerDelivery_004
[V2G2-577]	P	-	P	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-575]_[V2G2-577]
[V2G2-580]	-	-	P	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_MeteringReceipt_001
[V2G2-582]	-	P	-	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-582]
[V2G2-584]	-	P	-	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_CableCheck_001 TC_SECC_DC_VTB_CableCheck_002 TC_SECC_DC_VTB_CableCheck_006 TC_SECC_DC_VTB_CableCheck_007
[V2G2-587]	-	P	-	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_PreCharge_001
[V2G2-590]	-	P	-	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_PowerDelivery_001 TC_SECC_DC_VTB_PowerDelivery_012
[V2G2-593]	-	P	-	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-593]
[V2G2-597]	-	P	-	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_WeldingDetection_001

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-599]	-	-	-	-	-	P	-	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_CableCheck_001
[V2G2-601]	-	P	-	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-601]
[V2G2-602]	P	P	P	P	-	-	-	-	Table 7 can only be tested partially. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-603]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_SupportedAppProtocol_001
[V2G2-605]	P	P	P	P	P	P	P	P	Table 111 can only be tested partially Group_[V2G2-605]
[V2G2-606]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]
[V2G2-607]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]
[V2G2-608]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]
[V2G2-609]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]
[V2G2-610]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-611]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-612]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_([V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_([V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-613]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_([V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_([V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-614]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-617]	-	-	-	-	-	P	-	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_CableCheck_006
[V2G2-618]	-	-	-	-	-	P	-	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_PreCharge_006
[V2G2-619]	-	-	-	-	-	P	-	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-619]
[V2G2-620]	-	-	-	-	-	P	-	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_WeldingDetection_009
[V2G2-621]	-	P	-	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_CableCheck_006 TC_SECC_DC_VTB_CableCheck_007
[V2G2-623]	-	-	-	-	P	P	P	P	Table 16 can only be tested partially Group_[V2G2-142]_[V2G2-623]_[V2G2-018]_[V2G2-019]
[V2G2-624]	P	P	P	P	-	-	-	-	Table 16 can only be tested partially TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_007 TC_SECC_CMN_VTB_SDP_003
[V2G2-625]	X	X	-	-	-	-	-	-	TC_SECC_CMN_VTB_SDP_001 TC_SECC_CMN_VTB_SDP_002
[V2G2-626]	X	X	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_SDP_002 TC_SECC_CMN_VTB_SDP_003 TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-627]	X	X	-	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_CMN_VTB_SDP_007
[V2G2-628]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-166] TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-630]	-	-	-	-	-	-	X	X	TC_EVCC_CMN_VTB_SDP_004
[V2G2-631]	-	-	X	X	-	-	-	-	TC_SECC_CMN_VTB_SDP_003
[V2G2-632]	X	X	-	-	-	-	-	-	TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-633]	-	-	-	-	I	I	-	-	Requirement is only testable if a corresponding PIXIT is considered. Tested indirectly through test for requirement [V2G2-520], [V2G2-535] or [V2G2-619] Group_[V2G2-633]_[V2G2-635]_[V2G2-637]
[V2G2-634]	X	X	-	-	-	-	-	-	Group_([V2G2-634]
[V2G2-635]	-	-	-	-	I	I	-	-	Requirement is only testable if a corresponding PIXIT is considered. Tested indirectly through test for requirement [V2G2-520], [V2G2-535] or [V2G2-619] Group_[V2G2-633]_[V2G2-635]_[V2G2-637]
[V2G2-636]	X	X	-	-	-	-	-	-	TC_SECC_CMN_VTB_PaymentServiceSelection_001
[V2G2-637]	-	-	-	-	I	I	-	-	Requirement is only testable if a corresponding PIXIT is considered. Tested indirectly through test for requirement [V2G2-520], [V2G2-535] or [V2G2-619] Group_[V2G2-633]_[V2G2-635]_[V2G2-637]
[V2G2-645]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-166] TC_EVCC_CMN_VTB_SupportedAppProtocol_001 TC_EVCC_CMN_VTB_SupportedAppProtocol_004 TC_EVCC_CMN_VTB_SupportedAppProtocol_005
[V2G2-646]	-	-	-	-	X	X	-	-	Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_SupportedAppProtocol_004
[V2G2-652]	-	-	-	-	-	-	X	X	Group_[V2G2-006]_[V2G2-652]_[V2G2-909]
[V2G2-653]	-	-	P	P	-	-	P	P	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially Group_[V2G2-653]
[V2G2-656]	-	-	X	X	-	-	X	X	Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-659]	-	-	-	-	P	P	P	P	Table 104 can only be tested partially Group_[V2G2-659]_[V2G2-660]_[V2G2-661]_[V2G2-662]
[V2G2-660]	-	-	-	-	P	P	P	P	Table 104 can only be tested partially Group_[V2G2-659]_[V2G2-660]_[V2G2-661]_[V2G2-662]
[V2G2-661]	-	-	-	-	P	P	P	P	Table 104 can only be tested partially Group_[V2G2-659]_[V2G2-660]_[V2G2-661]_[V2G2-662]
[V2G2-662]	-	-	-	-	P	P	P	P	Table 104 can only be tested partially Group_[V2G2-659]_[V2G2-660]_[V2G2-661]_[V2G2-662]

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-664]	P	P	P	P	-	-	-	-	Table 104 can only be tested partially Group_[V2G2-664]_[V2G2-666]_[V2G2-667]
[V2G2-665]	P	P	P	P	-	-	-	-	Table 104 can only be tested partially Group_[V2G2-665]
[V2G2-666]	P	P	P	P	-	-	-	-	Table 104 can only be tested partially Group_[V2G2-664]_[V2G2-666]_[V2G2-667]
[V2G2-667]	P	P	P	P	-	-	-	-	Table 104 can only be tested partially Group_[V2G2-664]_[V2G2-666]_[V2G2-667]
[V2G2-672]	X	X	X	X	-	-	-	-	Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]
[V2G2-673]	-	-	-	-	X	X	X	X	Group_[V2G2-293]_[V2G2-673]
[V2G2-675]	-	-	-	-	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-675]
[V2G2-677]	P	P	P	P	-	-	-	-	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-677]
[V2G2-679]	-	-	-	-	X	X	X	X	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002
[V2G2-680]	-	-	-	-	X	X	X	X	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009
[V2G2-683]	-	-	-	-	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-683]
[V2G2-684]	-	-	-	-	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_CMN_VTB_Authorization_009 TC_EVCC_CMN_VTB_Authorization_001
[V2G2-685]	-	-	-	-	P	P	P	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_AC_VTB_ChargeParameterDiscovery_007 TC_EVCC_DC_VTB_ChargeParameterDiscovery_007
[V2G2-686]	-	-	-	-	-	P	-	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008
[V2G2-687]	P	P	P	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_Authorization_014 TC_SECC_CMN_VTB_Authorization_015 TC_SECC_CMN_VTB_Authorization_016

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_CMN_VTB_Authorization_017
[V2G2-688]	P	P	P	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_ChargeParameterDiscovery_014 TC_SECC_DC_VTB_ChargeParameterDiscovery_014
[V2G2-689]	-	-	-	-	X	-	X	-	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008 TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009
[V2G2-691]	X	-	-	-	-	-	-	-	TC_SECC_AC_VTB_ChargingStatus_001 TC_SECC_AC_VTB_ChargingStatus_006 TC_SECC_AC_VTB_ChargingStatus_008
[V2G2-696]	-	-	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-696]
[V2G2-700]	-	-	-	-	-	I	-	I	Tested indirectly through test for requirement [V2G2-702] TC_EVCC_DC_VTB_CableCheck_006
[V2G2-701]	-	-	-	-	-	I	-	I	Tested indirectly through test for requirement [V2G2-702] or [V2G2-525] TC_EVCC_DC_VTB_CableCheck_006
[V2G2-702]	-	-	-	-	-	P	-	P	Internal states and timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_CableCheck_006
[V2G2-704]	-	-	-	-	-	I	-	I	Tested indirectly through test for requirement [V2G2-706] TC_EVCC_DC_VTB_PreCharge_006
[V2G2-705]	-	-	-	-	-	I	-	I	Tested indirectly through test for requirement [V2G2-706] or [V2G2-528] Group_[V2G2-705]
[V2G2-706]	-	-	-	-	-	P	-	P	Internal states and timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_PreCharge_006
[V2G2-708]	X	-	X	-	-	-	-	-	Group_[V2G2-931]_[V2G2-708]
[V2G2-710]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-711] TC_EVCC_AC_VTB_ChargeParameterDiscovery_007 TC_EVCC_CMN_VTB_Authorization_009 TC_EVCC_DC_VTB_ChargeParameterDiscovery_007
[V2G2-711]	-	-	-	-	P	P	P	P	TC_EVCC_AC_VTB_ChargeParameterDiscovery_007 TC_EVCC_CMN_VTB_Authorization_009 TC_EVCC_DC_VTB_ChargeParameterDiscovery_007
[V2G2-712]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-713]. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. Group_[V2G2-712]_[V2G2-713]
[V2G2-713]	P	P	P	P	-	-	-	-	Internal states and timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									Group_[V2G2-712]_[V2G2-713]
[V2G2-714]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-716] TC_SECC_CMN_VTB_SessionSetup_004
[V2G2-715]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-716] TC_SECC_CMN_VTB_SessionSetup_004
[V2G2-716]	P	P	P	P	-	-	-	-	Internal states and timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_SessionSetup_004
[V2G2-717]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-717]
[V2G2-718]	-	-	-	-	P	P	P	P	Internal states cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially. Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-718]_[V2G2-739]
[V2G2-721]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-168] TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_002 TC_SECC_CMN_VTB_SupportedAppProtocol_006 TC_SECC_CMN_VTB_SupportedAppProtocol_004
[V2G2-723]	P	P	P	P	-	-	-	-	Internal states and timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_CMN_VTB_SDP_004 TC_SECC_CMN_VTB_SDP_005 TC_SECC_CMN_VTB_SDP_006
[V2G2-724]	X	X	X	X	-	-	-	-	TC_SECC_AC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_002
[V2G2-725]	P	P	P	P	-	-	-	-	Internal states cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially. (Requirements refers to ISO 15118-5) Group_[V2G2-725]
[V2G2-728]	-	-	-	-	X	X	X	X	Group_[V2G2-728]_[V2G2-025]
[V2G2-729]	X	X	X	X	-	-	-	-	Group_[V2G2-729]_[V2G2-034]
[V2G2-731]	-	-	-	-	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-158]_[V2G2-731]
[V2G2-733]	P	P	P	P	-	-	-	-	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-733]
[V2G2-736]	X	X	X	X	-	-	-	-	Group_[V2G2-539]_[V2G2-736]
[V2G2-737]	-	-	-	-	P	P	P	P	Requirement definition considers various possible scenarios, so the requirement can only be tested partially for exemplarily selected cases Group_[V2G2-737]
[V2G2-738]	N	N	N	N	X	X	X	X	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									Group_[V2G2-239]_[V2G2-738]
[V2G2-739]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-718]_[V2G2-739]
[V2G2-740]	-	-	-	-	P	P	P	P	Requirement is only testable if a corresponding PIXIT is considered. Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-740]
[V2G2-741]	P	P	P	P	-	-	-	-	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-741]_[V2G2-745]
[V2G2-742]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-742]
[V2G2-744]	-	-	-	-	I	I	I	I	Tested indirectly through test for requirement [V2G2-740]. Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-744]
[V2G2-745]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-741] Group_[V2G2-741]_[V2G2-745]
[V2G2-746]	-	-	-	-	X	X	X	X	TC_EVCC_CMN_VTB_SessionSetup_001 TC_EVCC_CMN_VTB_SessionSetup_008
[V2G2-747]	-	-	-	-	P	P	P	P	Only one message sequence will be considered, so the requirement can only be tested partially Group_[V2G2-747]
[V2G2-748]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-748]
[V2G2-750]	X	X	X	X	-	-	-	-	TC_SECC_AC_VTB_SessionSetup_003 TC_SECC_AC_VTB_SessionSetup_006 TC_SECC_CMN_VTB_SessionSetup_001 TC_SECC_DC_VTB_SessionSetup_003 TC_SECC_DC_VTB_SessionSetup_006
[V2G2-751]	P	P	P	P	X	X	X	X	Only one message sequence will be considered, so the requirement can only be tested partially TC_EVCC_CMN_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_002
[V2G2-752]	P	P	P	P	-	-	-	-	Only one message sequence will be considered, so the requirement can only be tested partially Group_[V2G2-752]
[V2G2-753]	I	I	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-754] TC_SECC_AC_VTB_SessionSetup_001 TC_SECC_AC_VTB_SessionSetup_004 TC_SECC_DC_VTB_SessionSetup_001 TC_SECC_DC_VTB_SessionSetup_004
[V2G2-754]	X	X	X	X	-	-	-	-	TC_SECC_AC_VTB_SessionSetup_001 TC_SECC_AC_VTB_SessionSetup_004 TC_SECC_DC_VTB_SessionSetup_001 TC_SECC_DC_VTB_SessionSetup_004
[V2G2-756]	X	X	X	X	-	-	-	-	TC_SECC_AC_VTB_SessionSetup_002

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_AC_VTB_SessionSetup_005 TC_SECC_DC_VTB_SessionSetup_002 TC_SECC_DC_VTB_SessionSetup_005
[V2G2-757]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-758]	X	X	X	X	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-759]	P	P	P	P	N	N	N	N	Cannot be tested explicitly in EVCC black box test configuration. Table 63 can only be tested partially TC_SECC_AC_VTB_ServiceDiscovery_001 TC_SECC_AC_VTB_ServiceDiscovery_002 TC_SECC_CMN_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_001 TC_SECC_DC_VTB_ServiceDiscovery_002
[V2G2-762]	X	X	-	-	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered Group_([V2G2-762]
[V2G2-764]	-	-	-	-	-	-	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-765]	-	-	-	-	-	-	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-766]	-	-	-	-	-	-	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-767]	-	-	-	-	-	-	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-768]	-	-	-	-	-	-	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-769]	-	-	-	-	-	-	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-770]	-	-	-	-	-	-	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-771]	-	-	-	-	-	-	X	X	Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]
[V2G2-773]	X	X	X	X	-	-	-	-	Group_[V2G2-773]
[V2G2-776]	-	-	-	-	-	-	X	X	Requirement is only testable if a corresponding PIXIT is considered Group_[V2G2-776]
[V2G2-778]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-779]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-780]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-781]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-782]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-783]	-	-	X	X	-	-	N	N	Cannot be tested explicitly in EVCC black box test configuration for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-787]	-	-	-	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_DC_VTB_CurrentDemand_002 TC_SECC_DC_VTB_CurrentDemand_007 TC_SECC_DC_VTB_CurrentDemand_009
[V2G2-788]	-	-	-	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered TC_SECC_DC_VTB_CurrentDemand_001 TC_SECC_DC_VTB_CurrentDemand_006 TC_SECC_DC_VTB_CurrentDemand_008

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-789]	-	-	-	-	-	-	-	X	Group_[V2G2-790]_[V2G2-789]
[V2G2-790]	-	-	-	-	-	-	-	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially Group_[V2G2-790]_[V2G2-789]
[V2G2-791]	-	-	-	-	-	-	-	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-791]
[V2G2-792]	-	-	-	-	-	-	-	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009 TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014
[V2G2-793]	-	-	-	-	-	-	-	P	Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002
[V2G2-794]	-	-	-	-	-	-	-	P	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in EVCC black box configuration, so the requirement can only be tested partially TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009
[V2G2-795]	-	-	-	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially Group_[V2G2-795]
[V2G2-797]	-	-	-	P	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_MeteringReceipt_001
[V2G2-799]	-	-	-	-	P	P	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-799]
[V2G2-800]	I	I	I	I	I	I	I	I	Tested indirectly through test for requirement [V2G2-160] Group_[V2G2-800]
[V2G2-804]	X	X	X	X	-	-	-	-	TC_SECC_CMN_VTB_PaymentServiceSelection_007
[V2G2-812]	P	P	P	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_014 TC_SECC_DC_VTB_PowerDelivery_013
[V2G2-813]	P	P	P	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_003 TC_SECC_DC_VTB_PowerDelivery_011
[V2G2-814]	-	-	X	X	-	-	-	-	TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-817]	-	-	-	-	-	-	I	I	Tested indirectly through test for requirement [V2G2-500]/[V2G2-501] and [V2G2-503]

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-818]	-	-	-	-	-	-	I	I	Tested indirectly through test for requirement [V2G2-500]/[V2G2-501] and [V2G2-503] TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-820]	-	-	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-554] and [V2G2-653] TC_SECC_CMN_VTB_Authorization_012 TC_SECC_CMN_VTB_Authorization_016
[V2G2-821]	-	-	I	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-557] and [V2G2-653] TC_SECC_CMN_VTB_Authorization_013 TC_SECC_CMN_VTB_Authorization_017
[V2G2-822]	-	-	P	P	-	-	P	P	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-827]	-	-	-	-	-	-	X	X	TC_EVCC_CMN_VTB_Authorization_011
[V2G2-828]	-	-	-	-	X	-	-	-	TC_EVCC_AC_VTB_PaymentServiceSelection_001
[V2G2-829]	-	-	-	-	X	-	X	-	Requirement is only testable if a corresponding PIXIT is considered. Group_[V2G2-829]
[V2G2-831]	P	P	P	P	-	-	P	P	Requirement is only testable if a corresponding PIXIT is considered. Table 66 can only be tested partially. Group_[V2G2-831]
[V2G2-832]	P	P	P	P	P	P	P	P	Table 68 can only be tested partially Group_[V2G2-832]
[V2G2-836]	-	-	-	-	X	X	X	X	Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]
[V2G2-839]	-	-	-	-	X	X	X	X	Group_[V2G2-839]
[V2G2-840]	-	-	-	-	X	X	X	X	Group_[V2G2-840]_[V2G2-841]
[V2G2-841]	-	-	-	-	I	I	I	I	Requirement is only testable if a corresponding PIXIT is considered. Tested indirectly through test for requirement [V2G2-840] Group_[V2G2-840]_[V2G2-841]
[V2G2-842]	-	-	-	-	X	X	X	X	Requirement is only testable if a corresponding PIXIT is considered. Group_[V2G2-842]
[V2G2-845]	-	-	-	-	X	X	-	-	TC_EVCC_CMN_VTB_Authorization_009
[V2G2-846]	-	-	-	-	X	-	X	-	Group_[V2G2-510]_[V2G2-847]_[V2G2-846]
[V2G2-847]	-	-	-	-	X	-	X	-	Group_[V2G2-510]_[V2G2-847]_[V2G2-846]
[V2G2-848]	-	-	-	-	X	-	X	-	Group_[V2G2-848]_[V2G2-849]
[V2G2-849]	-	-	-	-	I	-	I	-	Tested indirectly through test for requirement [V2G2-848] Group_[V2G2-848]_[V2G2-849]
[V2G2-852]	X	-	-	-	-	-	-	-	Group_([V2G2-852]

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
[V2G2-854]	X	X	-	-	-	-	-	-	Cannot be tested explicitly in SECC black box test configuration TC_SECC_CMN_VTB_Authorization_014
[V2G2-855]	-	-	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. TC_SECC_CMN_VTB_Authorization_015 TC_SECC_CMN_VTB_Authorization_016 TC_SECC_CMN_VTB_Authorization_017
[V2G2-856]	X	X	X	X	-	-	-	-	Group_[V2G2-563]_[V2G2-856]
[V2G2-861]	P	-	P	-	-	-	-	-	The contactor measuring cannot be tested in SECC black box test configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_001 TC_SECC_AC_VTB_PowerDelivery_002 TC_SECC_AC_VTB_PowerDelivery_003 TC_SECC_AC_VTB_PowerDelivery_004
[V2G2-864]	I	-	I	-	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_AC_VTB_PowerDelivery_005
[V2G2-871]	P	P	P	P	-	-	-	-	OCSP is out of scope, so the requirement can only be tested partially for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_SECC_CMN_VTB_SupportedAppProtocol_001 TC_SECC_CMN_VTB_SupportedAppProtocol_006
[V2G2-875]	-	-	-	-	P	P	P	P	OCSP is out of scope, so the requirement can only be tested partially for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered for the profile defined in the respective column. TC_EVCC_CMN_VTB_SupportedAppProtocol_001
[V2G2-879]	-	-	-	-	X	X	X	X	Has not been implemented yet due to a higher complexity TC_EVCC_CMN_VTB_SessionSetup_008
[V2G2-881]	-	P	-	P	-	-	-	-	Table 100 can only be tested partially Group_[V2G2-881]
[V2G2-884]	N	N	P	P	N	N	P	P	Cannot be tested explicitly in SECC/EVCC black box test configuration for the profile defined in the respective column. Only one case can be considered, so the requirement can only be tested partially for the profile defined in the respective column. TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001 TC_EVCC_CMN_VTB_CertificateUpdate_001 TC_EVCC_CMN_VTB_PaymentDetails_001 TC_SECC_CMN_VTB_CertificateInstallation_001 TC_SECC_CMN_VTB_CertificateUpdate_001
[V2G2-888]	-	-	-	-	-	-	X	X	TC_EVCC_CMN_VTB_CertificateUpdate_001
[V2G2-891]	-	-	-	-	-	-	P	P	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially Group_[V2G2-891]
[V2G2-893]	-	-	-	-	-	-	X	X	TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-896]	-	-	-	-	-	-	P	P	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									Group_[V2G2-896]
[V2G2-898]	-	-	-	-	-	-	X	X	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_001 TC_EVCC_CMN_VTB_PaymentDetails_002 TC_EVCC_CMN_VTB_PaymentDetails_003
[V2G2-899]	-	-	P	P	-	-	-	-	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially Group_[V2G2-899]
[V2G2-900]	-	-	-	-	-	-	X	X	TC_EVCC_CMN_VTB_Authorization_001 TC_EVCC_CMN_VTB_Authorization_010 TC_EVCC_CMN_VTB_Authorization_011
[V2G2-901]	-	-	P	P	-	-	-	-	The verification of this requirement consists of two different test cases, so the requirement can only be tested partially TC_SECC_CMN_VTB_Authorization_002 TC_SECC_CMN_VTB_Authorization_007 TC_SECC_CMN_VTB_Authorization_010 TC_SECC_CMN_VTB_Authorization_011 TC_SECC_CMN_VTB_Authorization_015
[V2G2-903]	-	-	-	-	-	-	X	X	Group_[V2G2-903]
[V2G2-904]	-	-	I	I	-	-	-	-	Tested indirectly through test for requirements [V2G2-580] or [V2G2-797] TC_SECC_AC_VTB_MeteringReceipt_001 TC_SECC_DC_VTB_MeteringReceipt_001
[V2G2-905]	N	N	I	I	-	-	-	-	Cannot be tested explicitly in SECC black box test configuration for the profile defined in the respective column. Tested indirectly through test of signature validation for the profile defined in the respective column. Requirement is only testable if a corresponding PIXIT is considered Group_([V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_([V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-906]	-	-	X	X	-	-	-	-	Requirement is only testable if a corresponding PIXIT is considered. Group_([V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_([V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]
[V2G2-909]	-	-	-	-	-	-	X	X	Group_[V2G2-006]_[V2G2-652]_[V2G2-909]
[V2G2-912]	-	-	-	-	-	X	-	X	TC_EVCC_DC_VTB_CableCheck_001
[V2G2-913]	-	-	-	-	-	X	-	X	Group_[V2G2-913]_[V2G2-914]
[V2G2-914]	-	-	-	-	-	I	-	I	Tested indirectly through test for requirement [V2G2-913] Group_[V2G2-913]_[V2G2-914]
[V2G2-915]	-	P	-	P	-	-	-	-	Only one case can be considered, so the requirement can only be tested partially Group_[V2G2-915]
[V2G2-917]	-	P	-	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_CableCheck_001 TC_SECC_DC_VTB_CableCheck_002

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Requirement ID	Covered in Test Suite								TC ID(s) / Comment
	SECC				EVCC				
	EIM		PnC		EIM		PnC		
	AC	DC	AC	DC	AC	DC	AC	DC	
									TC_SECC_DC_VTB_CableCheck_006 TC_SECC_DC_VTB_CableCheck_007
[V2G2-918]	-	X	-	X	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_CableCheck_008
[V2G2-919]	-	I	-	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-922] TC_SECC_DC_VTB_SessionStop_006 TC_SECC_DC_VTB_WeldingDetection_005
[V2G2-920]	-	I	-	I	-	-	-	-	Tested indirectly through test for requirement [V2G2-922] TC_SECC_DC_VTB_SessionStop_006 TC_SECC_DC_VTB_WeldingDetection_005
[V2G2-921]	-	P	-	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_SessionStop_001 TC_SECC_DC_VTB_SessionStop_002 TC_SECC_DC_VTB_WeldingDetection_001
[V2G2-922]	-	P	-	P	-	-	-	-	Internal timings cannot be tested explicitly in SECC black box configuration, so the requirement can only be tested partially TC_SECC_DC_VTB_SessionStop_006 TC_SECC_DC_VTB_WeldingDetection_005
[V2G2-926]	N	N	X	X	N	N	X	X	Cannot be tested explicitly in SECC/EVCC black box test configuration for the profile defined in the respective column. Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]
[V2G2-930]	-	-	-	-	I	-	I	-	Tested indirectly through the PTC for IEC 61851-1 signaling (see subsection 7.7.7) Group_[V2G2-930]
[V2G2-931]	I	-	I	-	-	-	-	-	Tested indirectly through the PTC for IEC 61851-1 signaling (see subsection 7.7.7) Group_[V2G2-931]_[V2G2-708]
[V2G2-932]	-	-	-	-	-	-	X	X	TC_EVCC_CMN_VTB_CertificateInstallation_001
[V2G2-933]	-	-	-	-	-	-	P	P	Only one field can be considered, so the requiremnent can only be tested partially TC_EVCC_CMN_VTB_CertificateInstallation_001

Several requirements in ISO 15118-2 are covered in multiple test cases of the ATS. For a simplified representation, the relevant TC identifier of each test case is mapped to the corresponding requirement. These special groups are summarized in Table 26. If two requirement groups refer to an identical set of TC identifiers, these groups are consolidated.

Table 26 — Groups for a simplified TC Id representation (see Table 25)

Group name	TC IDs
Group_[V2G2-159]	TC_EVCC_AC_VTB_SDP_001, TC_EVCC_AC_VTB_SDP_002, TC_EVCC_AC_VTB_SDP_003, TC_EVCC_AC_VTB_SDP_005, TC_EVCC_AC_VTB_SDP_006, TC_EVCC_AC_VTB_SDP_007, TC_EVCC_AC_VTB_SDP_009, TC_EVCC_AC_VTB_SDP_010,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_SDP_011, TC_EVCC_AC_VTB_SDP_013, TC_EVCC_AC_VTB_SDP_014, TC_EVCC_AC_VTB_SDP_015, TC_EVCC_AC_VTB_SDP_004, TC_EVCC_AC_VTB_SDP_008, TC_EVCC_AC_VTB_SDP_012, TC_EVCC_AC_VTB_SDP_016, TC_EVCC_CMN_VTB_SDP_005, TC_EVCC_CMN_VTB_SDP_006, TC_EVCC_DC_VTB_SDP_001, TC_EVCC_DC_VTB_SDP_002, TC_EVCC_DC_VTB_SDP_003, TC_EVCC_DC_VTB_SDP_005, TC_EVCC_DC_VTB_SDP_006, TC_EVCC_DC_VTB_SDP_007, TC_EVCC_DC_VTB_SDP_009, TC_EVCC_DC_VTB_SDP_010, TC_EVCC_DC_VTB_SDP_011, TC_EVCC_DC_VTB_SDP_013, TC_EVCC_DC_VTB_SDP_014, TC_EVCC_DC_VTB_SDP_015, TC_EVCC_DC_VTB_SDP_004, TC_EVCC_DC_VTB_SDP_008, TC_EVCC_DC_VTB_SDP_012, TC_EVCC_DC_VTB_SDP_016,
Group_[V2G2-747]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_015,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_Authorization_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_CMN_VTB_ServiceDetail_002, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_CMN_VTB_ServiceDetail_001, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007,
Group_[V2G2-376]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_CMN_VTB_SupportedAppProtocol_001, TC_EVCC_CMN_VTB_Authorization_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_CMN_VTB_SupportedAppProtocol_004, TC_EVCC_CMN_VTB_SupportedAppProtocol_005,
Group_[V2G2-380]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_012,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_CMN_VTB_SupportedAppProtocol_001, TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001,
Group_[V2G2-832]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreCharge_006, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012,
Group_[V2G2-505]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026,
Group_[V2G2-930]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026,
Group_[V2G2-215]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014,
Group_[V2G2-740]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007,
Group_[V2G2-744]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_005,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007,
Group_[V2G2-737]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_009, TC_EVCC_AC_VTB_ChargeParameterDiscovery_010, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012, TC_EVCC_AC_VTB_ChargingStatus_006, TC_EVCC_AC_VTB_ChargingStatus_007, TC_EVCC_AC_VTB_PowerDelivery_011, TC_EVCC_AC_VTB_PowerDelivery_012, TC_EVCC_AC_VTB_MeteringReceipt_011, TC_EVCC_AC_VTB_MeteringReceipt_012, TC_EVCC_CMN_VTB_Authorization_013, TC_EVCC_CMN_VTB_Authorization_014, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012, TC_EVCC_CMN_VTB_ServiceDiscovery_006, TC_EVCC_CMN_VTB_ServiceDiscovery_007, TC_EVCC_CMN_VTB_SessionSetup_006, TC_EVCC_CMN_VTB_SessionSetup_007, TC_EVCC_CMN_VTB_ServiceDetail_015, TC_EVCC_CMN_VTB_ServiceDetail_017, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008, TC_EVCC_CMN_VTB_CertificateInstallation_011, TC_EVCC_CMN_VTB_CertificateInstallation_012, TC_EVCC_CMN_VTB_CertificateUpdate_013, TC_EVCC_CMN_VTB_CertificateUpdate_014, TC_EVCC_CMN_VTB_ServiceDetail_016, TC_EVCC_CMN_VTB_ServiceDetail_018, TC_EVCC_DC_VTB_CableCheck_008, TC_EVCC_DC_VTB_CableCheck_009, TC_EVCC_DC_VTB_ChargeParameterDiscovery_009, TC_EVCC_DC_VTB_ChargeParameterDiscovery_010, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012, TC_EVCC_DC_VTB_CurrentDemand_006, TC_EVCC_DC_VTB_CurrentDemand_007, TC_EVCC_DC_VTB_PreCharge_008, TC_EVCC_DC_VTB_PreCharge_009, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012, TC_EVCC_DC_VTB_WeldingDetection_007, TC_EVCC_DC_VTB_WeldingDetection_008, TC_EVCC_DC_VTB_MeteringReceipt_011, TC_EVCC_DC_VTB_MeteringReceipt_012,
Group_[V2G2-799]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007, TC_EVCC_AC_VTB_PowerDelivery_002, TC_EVCC_AC_VTB_PowerDelivery_003, TC_EVCC_AC_VTB_PowerDelivery_004, TC_EVCC_AC_VTB_PowerDelivery_005, TC_EVCC_AC_VTB_PowerDelivery_006, TC_EVCC_AC_VTB_PowerDelivery_007, TC_EVCC_AC_VTB_PowerDelivery_008, TC_EVCC_AC_VTB_PowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009,
Group_[V2G2-402]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020,
Group_[V2G2-404]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021,
Group_[V2G2-182]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021,
Group_[V2G2-675]	TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-842]	TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-829]	TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033,
Group_[V2G2-296]	TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018,
Group_[V2G2-520]	TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020,
Group_[V2G2-508]	TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_010,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_029,
Group_[V2G2-242]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014,
Group_[V2G2-839]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002,
Group_[V2G2-517]	TC_EVCC_AC_VTB_MeteringReceipt_002, TC_EVCC_AC_VTB_MeteringReceipt_003, TC_EVCC_AC_VTB_MeteringReceipt_004, TC_EVCC_AC_VTB_MeteringReceipt_005, TC_EVCC_AC_VTB_MeteringReceipt_006, TC_EVCC_AC_VTB_MeteringReceipt_007,
Group_[V2G2-903]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-831]	TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_012, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_012,
Group_[V2G2-776]	TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-490]	TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_015,
Group_[V2G2-482]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010, TC_EVCC_AC_VTB_ChargingStatus_008, TC_EVCC_AC_VTB_PowerDelivery_010, TC_EVCC_AC_VTB_SessionStop_006, TC_EVCC_AC_VTB_MeteringReceipt_010, TC_EVCC_CMN_VTB_Authorization_012, TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010, TC_EVCC_CMN_VTB_ServiceDiscovery_008, TC_EVCC_CMN_VTB_SessionSetup_005, TC_EVCC_CMN_VTB_ServiceDetail_013, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006, TC_EVCC_CMN_VTB_CertificateInstallation_010, TC_EVCC_CMN_VTB_CertificateUpdate_012, TC_EVCC_CMN_VTB_ServiceDetail_014, TC_EVCC_DC_VTB_CableCheck_007, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010, TC_EVCC_DC_VTB_CurrentDemand_008, TC_EVCC_DC_VTB_PreCharge_007, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010, TC_EVCC_DC_VTB_WeldingDetection_006, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006, TC_EVCC_DC_VTB_MeteringReceipt_010,
Group_[V2G2-506]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_002, TC_EVCC_AC_VTB_ChargeParameterDiscovery_003, TC_EVCC_AC_VTB_ChargeParameterDiscovery_004, TC_EVCC_AC_VTB_ChargeParameterDiscovery_005, TC_EVCC_AC_VTB_ChargeParameterDiscovery_006, TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_002, TC_EVCC_DC_VTB_ChargeParameterDiscovery_003, TC_EVCC_DC_VTB_ChargeParameterDiscovery_004, TC_EVCC_DC_VTB_ChargeParameterDiscovery_005, TC_EVCC_DC_VTB_ChargeParameterDiscovery_006,
Group_[V2G2-492]	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009,
Group_[V2G2-390]	TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_CMN_VTB_SupportedAppProtocol_001, TC_EVCC_CMN_VTB_Authorization_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_CMN_VTB_SupportedAppProtocol_004, TC_EVCC_CMN_VTB_SupportedAppProtocol_005, TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-504]	TC_EVCC_CMN_VTB_Authorization_003, TC_EVCC_CMN_VTB_Authorization_004, TC_EVCC_CMN_VTB_Authorization_005, TC_EVCC_CMN_VTB_Authorization_006, TC_EVCC_CMN_VTB_Authorization_008, TC_EVCC_CMN_VTB_Authorization_007,
Group_[V2G2-394]	TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_CMN_VTB_SupportedAppProtocol_001, TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007,
Group_[V2G2-498]	TC_EVCC_CMN_VTB_CertificateInstallation_002, TC_EVCC_CMN_VTB_CertificateInstallation_003, TC_EVCC_CMN_VTB_CertificateInstallation_004, TC_EVCC_CMN_VTB_CertificateInstallation_005, TC_EVCC_CMN_VTB_CertificateInstallation_006, TC_EVCC_CMN_VTB_CertificateInstallation_007, TC_EVCC_CMN_VTB_CertificateInstallation_008, TC_EVCC_CMN_VTB_CertificateInstallation_009,
Group_[V2G2-499]	TC_EVCC_CMN_VTB_CertificateUpdate_002, TC_EVCC_CMN_VTB_CertificateUpdate_003, TC_EVCC_CMN_VTB_CertificateUpdate_004,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_CMN_VTB_CertificateUpdate_005, TC_EVCC_CMN_VTB_CertificateUpdate_006, TC_EVCC_CMN_VTB_CertificateUpdate_007, TC_EVCC_CMN_VTB_CertificateUpdate_008, TC_EVCC_CMN_VTB_CertificateUpdate_009, TC_EVCC_CMN_VTB_CertificateUpdate_010, TC_EVCC_CMN_VTB_CertificateUpdate_011,
Group_[V2G2-896]	TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-891]	TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_020,
Group_[V2G2-491]	TC_EVCC_CMN_VTB_ServiceDetail_003, TC_EVCC_CMN_VTB_ServiceDetail_005, TC_EVCC_CMN_VTB_ServiceDetail_007, TC_EVCC_CMN_VTB_ServiceDetail_009, TC_EVCC_CMN_VTB_ServiceDetail_011, TC_EVCC_CMN_VTB_ServiceDetail_004, TC_EVCC_CMN_VTB_ServiceDetail_006, TC_EVCC_CMN_VTB_ServiceDetail_008, TC_EVCC_CMN_VTB_ServiceDetail_010, TC_EVCC_CMN_VTB_ServiceDetail_012,
Group_[V2G2-683]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014,
Group_[V2G2-742]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026,
Group_[V2G2-403]	TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-405]	TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-705]	TC_EVCC_DC_VTB_PreCharge_006, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-528]	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-748]	TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004,
Group_[V2G2-535]	TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-507]	TC_EVCC_AC_VTB_SessionStop_002, TC_EVCC_AC_VTB_SessionStop_003, TC_EVCC_AC_VTB_SessionStop_004, TC_EVCC_AC_VTB_SessionStop_005, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005,
Group_[V2G2-717]	TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011,
Group_[V2G2-619]	TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_030,
Group_[V2G2-791]	TC_EVCC_DC_VTB_MeteringReceipt_002, TC_EVCC_DC_VTB_MeteringReceipt_003, TC_EVCC_DC_VTB_MeteringReceipt_004, TC_EVCC_DC_VTB_MeteringReceipt_005, TC_EVCC_DC_VTB_MeteringReceipt_006, TC_EVCC_DC_VTB_MeteringReceipt_007,
Group_[V2G2-800]	TC_EVCC_CMN_VTB_V2GTPSDP_002, TC_EVCC_CMN_VTB_V2GTPSDP_003, TC_EVCC_CMN_VTB_V2GTPSDP_004, TC_EVCC_CMN_VTB_V2GTPSDP_005, TC_EVCC_CMN_VTB_V2GTPSessionSetup_002,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_CMN_VTB_V2GTPSessionSetup_003, TC_EVCC_CMN_VTB_V2GTPSessionSetup_004, TC_EVCC_CMN_VTB_V2GTPSessionSetup_005, TC_SECC_CMN_VTB_V2GTPSDP_002, TC_SECC_CMN_VTB_V2GTPSDP_003, TC_SECC_CMN_VTB_V2GTPSDP_004, TC_SECC_CMN_VTB_V2GTPSDP_005, TC_SECC_CMN_VTB_V2GTPSessionSetup_002, TC_SECC_CMN_VTB_V2GTPSessionSetup_003, TC_SECC_CMN_VTB_V2GTPSessionSetup_004, TC_SECC_CMN_VTB_V2GTPSessionSetup_005,
Group_[V2G2-605]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_007, TC_EVCC_CMN_VTB_Authorization_009, TC_EVCC_DC_VTB_CableCheck_006, TC_EVCC_DC_VTB_ChargeParameterDiscovery_007, TC_EVCC_DC_VTB_PreCharge_006, TC_SECC_CMN_VTB_SDP_004, TC_SECC_CMN_VTB_SDP_005, TC_SECC_CMN_VTB_SDP_006, TC_SECC_CMN_VTB_SessionSetup_004, TC_SECC_CMN_VTB_SupportedAppProtocol_005,
Group_[V2G2-665]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_010, TC_SECC_AC_VTB_ChargingStatus_010, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_012, TC_SECC_AC_VTB_ChargingStatus_012, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_001,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_034,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012, TC_SECC_DC_VTB_CurrentDemand_012, TC_SECC_DC_VTB_MeteringReceipt_001, TC_SECC_DC_VTB_MeteringReceipt_001,
Group_[V2G2-752]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_004, TC_SECC_AC_VTB_ChargeParameterDiscovery_005, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_SessionStop_002, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_010, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_012, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_CMN_VTB_PaymentServiceSelection_006, TC_SECC_CMN_VTB_PaymentServiceSelection_007, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_003,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_009, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_ServiceDetail_007, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_CMN_VTB_Authorization_007, TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateInstallation_007, TC_SECC_CMN_VTB_CertificateInstallation_008, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006, TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009, TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentDetails_004, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_PaymentServiceSelection_010, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_CMN_VTB_ServiceDetail_008, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_CableCheck_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_003, TC_SECC_DC_VTB_ChargeParameterDiscovery_004, TC_SECC_DC_VTB_ChargeParameterDiscovery_005,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_DC_VTB_CurrentDemand_004, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_DC_VTB_PreCharge_003, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_SessionStop_007, TC_SECC_DC_VTB_SessionStop_008, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_WeldingDetection_002, TC_SECC_DC_VTB_WeldingDetection_003, TC_SECC_DC_VTB_WeldingDetection_005,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012, TC_SECC_DC_VTB_MeteringReceipt_001, TC_SECC_DC_VTB_MeteringReceipt_002, TC_SECC_DC_VTB_MeteringReceipt_003, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007,
Group_[V2G2-377]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_SessionSetup_001,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_SupportedAppProtocol_004,
Group_[V2G2-381]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_010, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_012, TC_SECC_AC_VTB_MeteringReceipt_001,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentServiceSelection_003,
Group_[V2G2-457]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_ChargingStatus_002,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_010, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_012, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012, TC_SECC_DC_VTB_MeteringReceipt_001,
Group_[V2G2-677]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_WeldingDetection_001,
Group_[V2G2-304]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_007,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-773]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_029,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-303]	TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-725]	TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006,
Group_[V2G2-733]	TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006,
Group_[V2G2-460]	TC_SECC_AC_VTB_ChargeParameterDiscovery_003, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_CMN_VTB_PaymentServiceSelection_005, TC_SECC_CMN_VTB_ServiceDiscovery_003, TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_ServiceDetail_005, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_CMN_VTB_CertificateInstallation_003, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_PaymentDetails_005, TC_SECC_CMN_VTB_ServiceDetail_006, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_DC_VTB_ChargeParameterDiscovery_002, TC_SECC_DC_VTB_CurrentDemand_004, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PreCharge_003, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_008, TC_SECC_DC_VTB_WeldingDetection_003, TC_SECC_DC_VTB_MeteringReceipt_003,
Group_[V2G2-435]	TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargingStatus_005, TC_SECC_AC_VTB_PowerDelivery_012, TC_SECC_AC_VTB_PowerDelivery_013, TC_SECC_AC_VTB_SessionStop_004, TC_SECC_AC_VTB_MeteringReceipt_005, TC_SECC_CMN_VTB_ChargeParameterDiscovery_002, TC_SECC_CMN_VTB_PaymentServiceSelection_008, TC_SECC_CMN_VTB_ServiceDiscovery_004,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_Authorization_008, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_ServiceDetail_009, TC_SECC_CMN_VTB_Authorization_009, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_CertificateInstallation_004, TC_SECC_CMN_VTB_CertificateUpdate_004, TC_SECC_CMN_VTB_PaymentDetails_006, TC_SECC_CMN_VTB_ServiceDetail_010, TC_SECC_DC_VTB_CableCheck_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_CurrentDemand_005, TC_SECC_DC_VTB_PowerDelivery_009, TC_SECC_DC_VTB_PowerDelivery_010, TC_SECC_DC_VTB_PreCharge_004, TC_SECC_DC_VTB_SessionStop_005, TC_SECC_DC_VTB_SessionStop_009, TC_SECC_DC_VTB_WeldingDetection_004, TC_SECC_DC_VTB_MeteringReceipt_005,
Group_[V2G2-915]	TC_SECC_CMN_VTB_SDP_001, TC_SECC_CMN_VTB_SDP_002, TC_SECC_CMN_VTB_SDP_007, TC_SECC_CMN_VTB_SDP_003, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012, TC_SECC_DC_VTB_MeteringReceipt_001,
Group_[V2G2-391]	TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_WeldingDetection_001,
Group_[V2G2-395]	TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_PaymentDetails_001,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012, TC_SECC_DC_VTB_MeteringReceipt_001,
Group_[V2G2-653]	TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_DC_VTB_MeteringReceipt_001,
Group_[V2G2-461]	TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006,
Group_[V2G2-696]	TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006, TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009, TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017,
Group_[V2G2-458]	TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_CMN_VTB_CertificateInstallation_008, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_017, TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_WeldingDetection_005,
Group_[V2G2-899]	TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentDetails_007, TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010,
Group_[V2G2-582]	TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-601]	TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021,
Group_[V2G2-881]	TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021,
Group_[V2G2-462]	TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006,
Group_[V2G2-593]	TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012,
Group_[V2G2-795]	TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012,
Group_[V2G2-481]	TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007,
Group_([V2G2-634]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_014,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_WeldingDetection_001,
Group_([V2G2-852]	TC_SECC_CMN_VTB_SDP_001, TC_SECC_CMN_VTB_SDP_002, TC_SECC_CMN_VTB_SDP_007, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_006,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_SupportedAppProtocol_004,
Group_([V2G2-762]	TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_015,
Group_[V2G2-158]_[V2G2-731]	TC_EVCC_AC_VTB_SDP_001, TC_EVCC_AC_VTB_SDP_002, TC_EVCC_AC_VTB_SDP_003, TC_EVCC_AC_VTB_SDP_005, TC_EVCC_AC_VTB_SDP_006, TC_EVCC_AC_VTB_SDP_007, TC_EVCC_AC_VTB_SDP_009, TC_EVCC_AC_VTB_SDP_010, TC_EVCC_AC_VTB_SDP_011, TC_EVCC_AC_VTB_SDP_013, TC_EVCC_AC_VTB_SDP_014, TC_EVCC_AC_VTB_SDP_015, TC_EVCC_AC_VTB_SDP_004, TC_EVCC_AC_VTB_SDP_008, TC_EVCC_AC_VTB_SDP_012, TC_EVCC_AC_VTB_SDP_016, TC_EVCC_DC_VTB_SDP_001, TC_EVCC_DC_VTB_SDP_002, TC_EVCC_DC_VTB_SDP_003, TC_EVCC_DC_VTB_SDP_005, TC_EVCC_DC_VTB_SDP_006, TC_EVCC_DC_VTB_SDP_007, TC_EVCC_DC_VTB_SDP_009, TC_EVCC_DC_VTB_SDP_010, TC_EVCC_DC_VTB_SDP_011, TC_EVCC_DC_VTB_SDP_013, TC_EVCC_DC_VTB_SDP_014, TC_EVCC_DC_VTB_SDP_015, TC_EVCC_DC_VTB_SDP_004, TC_EVCC_DC_VTB_SDP_008, TC_EVCC_DC_VTB_SDP_012, TC_EVCC_DC_VTB_SDP_016,
Group_[V2G2-728]_[V2G2-025]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_002, TC_EVCC_AC_VTB_ChargeParameterDiscovery_003, TC_EVCC_AC_VTB_ChargeParameterDiscovery_004, TC_EVCC_AC_VTB_ChargeParameterDiscovery_005, TC_EVCC_AC_VTB_ChargeParameterDiscovery_006, TC_EVCC_AC_VTB_ChargeParameterDiscovery_007, TC_EVCC_AC_VTB_ChargeParameterDiscovery_009, TC_EVCC_AC_VTB_ChargeParameterDiscovery_010, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012, TC_EVCC_AC_VTB_ChargingStatus_002, TC_EVCC_AC_VTB_ChargingStatus_003, TC_EVCC_AC_VTB_ChargingStatus_004, TC_EVCC_AC_VTB_ChargingStatus_005, TC_EVCC_AC_VTB_ChargingStatus_006, TC_EVCC_AC_VTB_ChargingStatus_007, TC_EVCC_AC_VTB_ChargingStatus_008, TC_EVCC_AC_VTB_PowerDelivery_002, TC_EVCC_AC_VTB_PowerDelivery_003, TC_EVCC_AC_VTB_PowerDelivery_004, TC_EVCC_AC_VTB_PowerDelivery_005, TC_EVCC_AC_VTB_PowerDelivery_006, TC_EVCC_AC_VTB_PowerDelivery_007, TC_EVCC_AC_VTB_PowerDelivery_008, TC_EVCC_AC_VTB_PowerDelivery_009, TC_EVCC_AC_VTB_PowerDelivery_010, TC_EVCC_AC_VTB_PowerDelivery_011, TC_EVCC_AC_VTB_PowerDelivery_012, TC_EVCC_AC_VTB_SessionStop_002,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_SessionStop_003, TC_EVCC_AC_VTB_SessionStop_004, TC_EVCC_AC_VTB_SessionStop_005, TC_EVCC_AC_VTB_SessionStop_006, TC_EVCC_AC_VTB_MeteringReceipt_002, TC_EVCC_AC_VTB_MeteringReceipt_003, TC_EVCC_AC_VTB_MeteringReceipt_004, TC_EVCC_AC_VTB_MeteringReceipt_005, TC_EVCC_AC_VTB_MeteringReceipt_006, TC_EVCC_AC_VTB_MeteringReceipt_007, TC_EVCC_AC_VTB_MeteringReceipt_010, TC_EVCC_AC_VTB_MeteringReceipt_011, TC_EVCC_AC_VTB_MeteringReceipt_012, TC_EVCC_CMN_VTB_Authorization_003, TC_EVCC_CMN_VTB_Authorization_004, TC_EVCC_CMN_VTB_Authorization_005, TC_EVCC_CMN_VTB_Authorization_006, TC_EVCC_CMN_VTB_Authorization_008, TC_EVCC_CMN_VTB_Authorization_009, TC_EVCC_CMN_VTB_Authorization_012, TC_EVCC_CMN_VTB_Authorization_013, TC_EVCC_CMN_VTB_Authorization_014, TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001, TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012, TC_EVCC_CMN_VTB_ServiceDiscovery_002, TC_EVCC_CMN_VTB_ServiceDiscovery_003, TC_EVCC_CMN_VTB_ServiceDiscovery_004, TC_EVCC_CMN_VTB_ServiceDiscovery_005, TC_EVCC_CMN_VTB_ServiceDiscovery_006, TC_EVCC_CMN_VTB_ServiceDiscovery_007, TC_EVCC_CMN_VTB_ServiceDiscovery_008, TC_EVCC_CMN_VTB_SessionSetup_002, TC_EVCC_CMN_VTB_SessionSetup_003, TC_EVCC_CMN_VTB_SessionSetup_004, TC_EVCC_CMN_VTB_SessionSetup_005, TC_EVCC_CMN_VTB_SessionSetup_006, TC_EVCC_CMN_VTB_SessionSetup_007, TC_EVCC_CMN_VTB_SupportedAppProtocol_002, TC_EVCC_CMN_VTB_SupportedAppProtocol_003, TC_EVCC_CMN_VTB_ServiceDetail_003, TC_EVCC_CMN_VTB_ServiceDetail_005, TC_EVCC_CMN_VTB_ServiceDetail_007, TC_EVCC_CMN_VTB_ServiceDetail_009, TC_EVCC_CMN_VTB_ServiceDetail_011, TC_EVCC_CMN_VTB_ServiceDetail_013, TC_EVCC_CMN_VTB_ServiceDetail_015, TC_EVCC_CMN_VTB_ServiceDetail_017, TC_EVCC_CMN_VTB_Authorization_007, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008, TC_EVCC_CMN_VTB_CertificateInstallation_002, TC_EVCC_CMN_VTB_CertificateInstallation_003, TC_EVCC_CMN_VTB_CertificateInstallation_004, TC_EVCC_CMN_VTB_CertificateInstallation_005, TC_EVCC_CMN_VTB_CertificateInstallation_006, TC_EVCC_CMN_VTB_CertificateInstallation_007, TC_EVCC_CMN_VTB_CertificateInstallation_008, TC_EVCC_CMN_VTB_CertificateInstallation_009, TC_EVCC_CMN_VTB_CertificateInstallation_010, TC_EVCC_CMN_VTB_CertificateInstallation_011, TC_EVCC_CMN_VTB_CertificateInstallation_012, TC_EVCC_CMN_VTB_CertificateUpdate_002, TC_EVCC_CMN_VTB_CertificateUpdate_003, TC_EVCC_CMN_VTB_CertificateUpdate_004, TC_EVCC_CMN_VTB_CertificateUpdate_005, TC_EVCC_CMN_VTB_CertificateUpdate_006, TC_EVCC_CMN_VTB_CertificateUpdate_007, TC_EVCC_CMN_VTB_CertificateUpdate_008, TC_EVCC_CMN_VTB_CertificateUpdate_009, TC_EVCC_CMN_VTB_CertificateUpdate_010, TC_EVCC_CMN_VTB_CertificateUpdate_011, TC_EVCC_CMN_VTB_CertificateUpdate_012, TC_EVCC_CMN_VTB_CertificateUpdate_013, TC_EVCC_CMN_VTB_CertificateUpdate_014, TC_EVCC_CMN_VTB_ServiceDetail_004, TC_EVCC_CMN_VTB_ServiceDetail_006, TC_EVCC_CMN_VTB_ServiceDetail_008, TC_EVCC_CMN_VTB_ServiceDetail_010, TC_EVCC_CMN_VTB_ServiceDetail_012, TC_EVCC_CMN_VTB_ServiceDetail_014, TC_EVCC_CMN_VTB_ServiceDetail_016, TC_EVCC_CMN_VTB_ServiceDetail_018, TC_EVCC_DC_VTB_CableCheck_002, TC_EVCC_DC_VTB_CableCheck_003, TC_EVCC_DC_VTB_CableCheck_004, TC_EVCC_DC_VTB_CableCheck_005, TC_EVCC_DC_VTB_CableCheck_006, TC_EVCC_DC_VTB_CableCheck_007, TC_EVCC_DC_VTB_CableCheck_008, TC_EVCC_DC_VTB_CableCheck_009, TC_EVCC_DC_VTB_ChargeParameterDiscovery_002, TC_EVCC_DC_VTB_ChargeParameterDiscovery_003, TC_EVCC_DC_VTB_ChargeParameterDiscovery_004, TC_EVCC_DC_VTB_ChargeParameterDiscovery_005, TC_EVCC_DC_VTB_ChargeParameterDiscovery_006, TC_EVCC_DC_VTB_ChargeParameterDiscovery_007, TC_EVCC_DC_VTB_ChargeParameterDiscovery_009, TC_EVCC_DC_VTB_ChargeParameterDiscovery_010, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012, TC_EVCC_DC_VTB_CurrentDemand_002, TC_EVCC_DC_VTB_CurrentDemand_003, TC_EVCC_DC_VTB_CurrentDemand_004, TC_EVCC_DC_VTB_CurrentDemand_005, TC_EVCC_DC_VTB_CurrentDemand_006, TC_EVCC_DC_VTB_CurrentDemand_007, TC_EVCC_DC_VTB_CurrentDemand_008, TC_EVCC_DC_VTB_PreCharge_002, TC_EVCC_DC_VTB_PreCharge_003, TC_EVCC_DC_VTB_PreCharge_004, TC_EVCC_DC_VTB_PreCharge_005, TC_EVCC_DC_VTB_PreCharge_006, TC_EVCC_DC_VTB_PreCharge_007, TC_EVCC_DC_VTB_PreCharge_008, TC_EVCC_DC_VTB_PreCharge_009, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012, TC_EVCC_DC_VTB_WeldingDetection_002, TC_EVCC_DC_VTB_WeldingDetection_003, TC_EVCC_DC_VTB_WeldingDetection_004, TC_EVCC_DC_VTB_WeldingDetection_005, TC_EVCC_DC_VTB_WeldingDetection_006, TC_EVCC_DC_VTB_WeldingDetection_007, TC_EVCC_DC_VTB_WeldingDetection_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006, TC_EVCC_DC_VTB_MeteringReceipt_002, TC_EVCC_DC_VTB_MeteringReceipt_003, TC_EVCC_DC_VTB_MeteringReceipt_004, TC_EVCC_DC_VTB_MeteringReceipt_005, TC_EVCC_DC_VTB_MeteringReceipt_006, TC_EVCC_DC_VTB_MeteringReceipt_007, TC_EVCC_DC_VTB_MeteringReceipt_010, TC_EVCC_DC_VTB_MeteringReceipt_011, TC_EVCC_DC_VTB_MeteringReceipt_012,
Group_[V2G2-214]_[V2G2-216]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026,
Group_[V2G2-278]_[V2G2-279]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_010, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012,
Group_[V2G2-360]_[V2G2-361]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026,
Group_[V2G2-293]_[V2G2-673]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002,
Group_[V2G2-297]_[V2G2-298]	TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020,
Group_[V2G2-239]_[V2G2-738]	TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-718]_[V2G2-739]	TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-848]_[V2G2-849]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014,
Group_[V2G2-245]_[V2G2-246]	TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-276]_[V2G2-277]	TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_012, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012,
Group_[V2G2-512]_[V2G2-408]	TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021,
Group_[V2G2-005]_[V2G2-007]	TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_MeteringReceipt_021, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_DC_VTB_MeteringReceipt_001,
Group_[V2G2-370]_[V2G2-371]	TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026,
Group_[V2G2-840]_[V2G2-841]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014,
Group_[V2G2-374]_[V2G2-375]	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014,
Group_[V2G2-253]_[V2G2-254]	TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-913]_[V2G2-914]	TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020,
Group_[V2G2-257]_[V2G2-258]	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014,
Group_[V2G2-790]_[V2G2-789]	TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-082]_[V2G2-086]	TC_EVCC_CMN_VTB_V2GTPSDP_001, TC_EVCC_CMN_VTB_V2GTPSDP_002, TC_EVCC_CMN_VTB_V2GTPSDP_003, TC_EVCC_CMN_VTB_V2GTPSDP_004, TC_EVCC_CMN_VTB_V2GTPSDP_005, TC_EVCC_CMN_VTB_V2GTPSessionSetup_001, TC_EVCC_CMN_VTB_V2GTPSessionSetup_002, TC_EVCC_CMN_VTB_V2GTPSessionSetup_003, TC_EVCC_CMN_VTB_V2GTPSessionSetup_004, TC_EVCC_CMN_VTB_V2GTPSessionSetup_005, TC_SECC_CMN_VTB_V2GTPSDP_001, TC_SECC_CMN_VTB_V2GTPSDP_002, TC_SECC_CMN_VTB_V2GTPSDP_003, TC_SECC_CMN_VTB_V2GTPSDP_004, TC_SECC_CMN_VTB_V2GTPSDP_005, TC_SECC_CMN_VTB_V2GTPSessionSetup_001, TC_SECC_CMN_VTB_V2GTPSessionSetup_002, TC_SECC_CMN_VTB_V2GTPSessionSetup_003, TC_SECC_CMN_VTB_V2GTPSessionSetup_004, TC_SECC_CMN_VTB_V2GTPSessionSetup_005,
Group_[V2G2-218]_[V2G2-220]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_003, TC_SECC_AC_VTB_ChargeParameterDiscovery_004, TC_SECC_AC_VTB_ChargeParameterDiscovery_005, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_002, TC_SECC_DC_VTB_ChargeParameterDiscovery_003,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_004, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-358]_[V2G2-359]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_010, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_012, TC_SECC_AC_VTB_MeteringReceipt_001,
Group_[V2G2-931]_[V2G2-708]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016,
Group_[V2G2-741]_[V2G2-745]	TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_004,
Group_[V2G2-729]_[V2G2-034]	TC_SECC_AC_VTB_ChargeParameterDiscovery_003, TC_SECC_AC_VTB_ChargeParameterDiscovery_004, TC_SECC_AC_VTB_ChargeParameterDiscovery_005, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_AC_VTB_ChargingStatus_005, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_012, TC_SECC_AC_VTB_PowerDelivery_013, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_AC_VTB_SessionStop_002, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_AC_VTB_SessionStop_004, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_005, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_CMN_VTB_ChargeParameterDiscovery_002, TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_CMN_VTB_PaymentServiceSelection_005, TC_SECC_CMN_VTB_PaymentServiceSelection_006, TC_SECC_CMN_VTB_PaymentServiceSelection_007, TC_SECC_CMN_VTB_PaymentServiceSelection_008, TC_SECC_CMN_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_ServiceDiscovery_003, TC_SECC_CMN_VTB_ServiceDiscovery_004, TC_SECC_CMN_VTB_SessionSetup_002, TC_SECC_CMN_VTB_SessionSetup_004, TC_SECC_CMN_VTB_SupportedAppProtocol_002, TC_SECC_CMN_VTB_Authorization_003, TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_Authorization_008, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_PaymentServiceSelection_009, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_ServiceDetail_005, TC_SECC_CMN_VTB_ServiceDetail_007, TC_SECC_CMN_VTB_ServiceDetail_009, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_CMN_VTB_Authorization_007, TC_SECC_CMN_VTB_Authorization_009,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateInstallation_003, TC_SECC_CMN_VTB_CertificateInstallation_004, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateInstallation_007, TC_SECC_CMN_VTB_CertificateInstallation_008, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_004, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006, TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009, TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017, TC_SECC_CMN_VTB_PaymentDetails_004, TC_SECC_CMN_VTB_PaymentDetails_005, TC_SECC_CMN_VTB_PaymentDetails_006, TC_SECC_CMN_VTB_PaymentDetails_007, TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010, TC_SECC_CMN_VTB_PaymentServiceSelection_010, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_CMN_VTB_ServiceDetail_006, TC_SECC_CMN_VTB_ServiceDetail_008, TC_SECC_CMN_VTB_ServiceDetail_010, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_DC_VTB_CableCheck_005, TC_SECC_DC_VTB_CableCheck_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_002, TC_SECC_DC_VTB_ChargeParameterDiscovery_003, TC_SECC_DC_VTB_ChargeParameterDiscovery_004, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_DC_VTB_CurrentDemand_004, TC_SECC_DC_VTB_CurrentDemand_005, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_009, TC_SECC_DC_VTB_PowerDelivery_010, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_DC_VTB_PreCharge_003, TC_SECC_DC_VTB_PreCharge_004, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_005, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_SessionStop_007, TC_SECC_DC_VTB_SessionStop_008,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_SessionStop_009, TC_SECC_DC_VTB_WeldingDetection_002, TC_SECC_DC_VTB_WeldingDetection_003, TC_SECC_DC_VTB_WeldingDetection_004, TC_SECC_DC_VTB_WeldingDetection_005, TC_SECC_DC_VTB_MeteringReceipt_002, TC_SECC_DC_VTB_MeteringReceipt_003, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_005, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007,
Group_[V2G2-539]_[V2G2-736]	TC_SECC_AC_VTB_ChargeParameterDiscovery_003, TC_SECC_AC_VTB_ChargeParameterDiscovery_004, TC_SECC_AC_VTB_ChargeParameterDiscovery_005, TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_AC_VTB_SessionStop_002, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_CMN_VTB_PaymentServiceSelection_005, TC_SECC_CMN_VTB_PaymentServiceSelection_006, TC_SECC_CMN_VTB_PaymentServiceSelection_007, TC_SECC_CMN_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_ServiceDiscovery_003, TC_SECC_CMN_VTB_SessionSetup_002, TC_SECC_CMN_VTB_SupportedAppProtocol_002, TC_SECC_CMN_VTB_Authorization_003, TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_PaymentServiceSelection_009, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_ServiceDetail_005, TC_SECC_CMN_VTB_ServiceDetail_007, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_CMN_VTB_Authorization_007, TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateInstallation_003, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateInstallation_007, TC_SECC_CMN_VTB_CertificateInstallation_008, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006, TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017, TC_SECC_CMN_VTB_PaymentDetails_004, TC_SECC_CMN_VTB_PaymentDetails_005, TC_SECC_CMN_VTB_PaymentDetails_007, TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010, TC_SECC_CMN_VTB_PaymentServiceSelection_010, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_CMN_VTB_ServiceDetail_006, TC_SECC_CMN_VTB_ServiceDetail_008, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_DC_VTB_CableCheck_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_002, TC_SECC_DC_VTB_ChargeParameterDiscovery_003, TC_SECC_DC_VTB_ChargeParameterDiscovery_004, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_DC_VTB_CurrentDemand_004, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_DC_VTB_PreCharge_003, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_SessionStop_007, TC_SECC_DC_VTB_SessionStop_008, TC_SECC_DC_VTB_WeldingDetection_002, TC_SECC_DC_VTB_WeldingDetection_003, TC_SECC_DC_VTB_WeldingDetection_005, TC_SECC_DC_VTB_MeteringReceipt_002, TC_SECC_DC_VTB_MeteringReceipt_003, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007,
Group_[V2G2-223]_[V2G2-226]	TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006, TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021,
Group_[V2G2-195]_[V2G2-196]	TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_SessionStop_002, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_ServiceDiscovery_003, TC_SECC_CMN_VTB_SessionSetup_002, TC_SECC_CMN_VTB_Authorization_003, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_PaymentDetails_004, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_007, TC_SECC_DC_VTB_WeldingDetection_002, TC_SECC_DC_VTB_MeteringReceipt_002,
Group_[V2G2-243]_[V2G2-244]	TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_010, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_012,
Group_[V2G2-575]_[V2G2-577]	TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_010, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_012,
Group_[V2G2-459]_[V2G2-538]	TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_SessionStop_002, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_CMN_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_SessionSetup_002, TC_SECC_CMN_VTB_Authorization_003, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_PaymentDetails_004, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_007,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_WeldingDetection_002, TC_SECC_DC_VTB_MeteringReceipt_002,
Group_[V2G2-165]_[V2G2-178]	TC_EVCC_CMN_VTB_SupportedAppProtocol_001, TC_EVCC_CMN_VTB_SupportedAppProtocol_004, TC_EVCC_CMN_VTB_SupportedAppProtocol_005, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_002, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_CMN_VTB_SupportedAppProtocol_004,
Group_[V2G2-203]_[V2G2-204]	TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_005, TC_SECC_CMN_VTB_PaymentServiceSelection_006, TC_SECC_CMN_VTB_PaymentServiceSelection_007, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_009, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_PaymentServiceSelection_010,
Group_[V2G2-212]_[V2G2-213]	TC_SECC_CMN_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_CMN_VTB_Authorization_007, TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017,
Group_[V2G2-563]_[V2G2-856]	TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017,
Group_[V2G2-103]_[V2G2-104]	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001,
Group_[V2G2-237]_[V2G2-238]	TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateInstallation_003, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateInstallation_007, TC_SECC_CMN_VTB_CertificateInstallation_008,
Group_[V2G2-231]_[V2G2-232]	TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009, TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017,
Group_[V2G2-208]_[V2G2-209]	TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentDetails_005, TC_SECC_CMN_VTB_PaymentDetails_007, TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010,
Group_[V2G2-199]_[V2G2-200]	TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_005, TC_SECC_CMN_VTB_ServiceDetail_007, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_006, TC_SECC_CMN_VTB_ServiceDetail_008, TC_SECC_CMN_VTB_ServiceDetail_012,
Group_[V2G2-251]_[V2G2-252]	TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_CableCheck_008,
Group_[V2G2-372]_[V2G2-373]	TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-712]_[V2G2-713]	TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_Authorization_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_014,
Group_[V2G2-240]_[V2G2-241]	TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_SessionStop_008,
Group_[V2G2-259]_[V2G2-260]	TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_004, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012,
Group_[V2G2-247]_[V2G2-248]	TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_DC_VTB_MeteringReceipt_001, TC_SECC_DC_VTB_MeteringReceipt_003, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007,
Group_[V2G2-157]_[V2G2-160]_[V2G2-161]	TC_EVCC_AC_VTB_SDP_001, TC_EVCC_AC_VTB_SDP_002, TC_EVCC_AC_VTB_SDP_003, TC_EVCC_AC_VTB_SDP_005, TC_EVCC_AC_VTB_SDP_006, TC_EVCC_AC_VTB_SDP_007, TC_EVCC_AC_VTB_SDP_009, TC_EVCC_AC_VTB_SDP_010, TC_EVCC_AC_VTB_SDP_011, TC_EVCC_AC_VTB_SDP_013, TC_EVCC_AC_VTB_SDP_014, TC_EVCC_AC_VTB_SDP_015, TC_EVCC_AC_VTB_SDP_004, TC_EVCC_AC_VTB_SDP_008, TC_EVCC_AC_VTB_SDP_012, TC_EVCC_AC_VTB_SDP_016, TC_EVCC_CMN_VTB_SDP_005, TC_EVCC_DC_VTB_SDP_001, TC_EVCC_DC_VTB_SDP_002, TC_EVCC_DC_VTB_SDP_003, TC_EVCC_DC_VTB_SDP_005, TC_EVCC_DC_VTB_SDP_006, TC_EVCC_DC_VTB_SDP_007, TC_EVCC_DC_VTB_SDP_009, TC_EVCC_DC_VTB_SDP_010, TC_EVCC_DC_VTB_SDP_011, TC_EVCC_DC_VTB_SDP_013, TC_EVCC_DC_VTB_SDP_014, TC_EVCC_DC_VTB_SDP_015, TC_EVCC_DC_VTB_SDP_004, TC_EVCC_DC_VTB_SDP_008, TC_EVCC_DC_VTB_SDP_012, TC_EVCC_DC_VTB_SDP_016,
Group_[V2G2-510]_[V2G2-847]_[V2G2-846]	TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021,
Group_[V2G2-006]_[V2G2-652]_[V2G2-909]	TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_[V2G2-493]_[V2G2-494]_[V2G2-432]	TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_013,
Group_[V2G2-188]_[V2G2-189]_[V2G2-485]	TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004,
Group_[V2G2-367]_[V2G2-368]_[V2G2-369]	TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014,
Group_[V2G2-633]_[V2G2-635]_[V2G2-637]	TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019,
Group_[V2G2-573]_[V2G2-362]_[V2G2-363]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-664]_[V2G2-666]_[V2G2-667]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_010, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_012, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SupportedAppProtocol_001, TC_SECC_CMN_VTB_SupportedAppProtocol_006, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_SupportedAppProtocol_004, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012, TC_SECC_DC_VTB_MeteringReceipt_001,
Group_[V2G2-441]_[V2G2-442]_[V2G2-443]	TC_SECC_AC_VTB_ChargingStatus_005, TC_SECC_AC_VTB_PowerDelivery_012, TC_SECC_AC_VTB_PowerDelivery_013, TC_SECC_AC_VTB_SessionStop_004, TC_SECC_AC_VTB_MeteringReceipt_005, TC_SECC_CMN_VTB_ChargeParameterDiscovery_002, TC_SECC_CMN_VTB_PaymentServiceSelection_008, TC_SECC_CMN_VTB_ServiceDiscovery_004, TC_SECC_CMN_VTB_Authorization_008, TC_SECC_CMN_VTB_ServiceDetail_009, TC_SECC_CMN_VTB_Authorization_009, TC_SECC_CMN_VTB_CertificateInstallation_004, TC_SECC_CMN_VTB_CertificateUpdate_004, TC_SECC_CMN_VTB_PaymentDetails_006, TC_SECC_CMN_VTB_ServiceDetail_010, TC_SECC_DC_VTB_CableCheck_005, TC_SECC_DC_VTB_CurrentDemand_005, TC_SECC_DC_VTB_PowerDelivery_009, TC_SECC_DC_VTB_PowerDelivery_010, TC_SECC_DC_VTB_PreCharge_004, TC_SECC_DC_VTB_SessionStop_005, TC_SECC_DC_VTB_SessionStop_009, TC_SECC_DC_VTB_WeldingDetection_004, TC_SECC_DC_VTB_MeteringReceipt_005,
Group_[V2G2-364]_[V2G2-365]_[V2G2-366]	TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012,
Group_[V2G2-142]_[V2G2-623]_[V2G2-018]_[V2G2-019]	TC_EVCC_AC_VTB_SDP_001, TC_EVCC_AC_VTB_SDP_002, TC_EVCC_AC_VTB_SDP_003, TC_EVCC_AC_VTB_SDP_005, TC_EVCC_AC_VTB_SDP_006, TC_EVCC_AC_VTB_SDP_007, TC_EVCC_AC_VTB_SDP_009, TC_EVCC_AC_VTB_SDP_010, TC_EVCC_AC_VTB_SDP_011, TC_EVCC_AC_VTB_SDP_013, TC_EVCC_AC_VTB_SDP_014, TC_EVCC_AC_VTB_SDP_015, TC_EVCC_AC_VTB_SDP_004, TC_EVCC_AC_VTB_SDP_008, TC_EVCC_AC_VTB_SDP_012, TC_EVCC_AC_VTB_SDP_016, TC_EVCC_CMN_VTB_SDP_005, TC_EVCC_CMN_VTB_SDP_006, TC_EVCC_CMN_VTB_SDP_001, TC_EVCC_CMN_VTB_SDP_002, TC_EVCC_CMN_VTB_SDP_003, TC_EVCC_CMN_VTB_SDP_004, TC_EVCC_DC_VTB_SDP_001, TC_EVCC_DC_VTB_SDP_002, TC_EVCC_DC_VTB_SDP_003, TC_EVCC_DC_VTB_SDP_005, TC_EVCC_DC_VTB_SDP_006, TC_EVCC_DC_VTB_SDP_007,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_SDP_009, TC_EVCC_DC_VTB_SDP_010, TC_EVCC_DC_VTB_SDP_011, TC_EVCC_DC_VTB_SDP_013, TC_EVCC_DC_VTB_SDP_014, TC_EVCC_DC_VTB_SDP_015, TC_EVCC_DC_VTB_SDP_004, TC_EVCC_DC_VTB_SDP_008, TC_EVCC_DC_VTB_SDP_012, TC_EVCC_DC_VTB_SDP_016,
Group_[V2G2-179]_[V2G2-180]_[V2G2-181]_[V2G2-183]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_CMN_VTB_Authorization_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_CMN_VTB_ServiceDetail_002, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_CMN_VTB_ServiceDetail_001, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_003, TC_SECC_AC_VTB_ChargeParameterDiscovery_004, TC_SECC_AC_VTB_ChargeParameterDiscovery_005, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_AC_VTB_ChargingStatus_001, TC_SECC_AC_VTB_ChargingStatus_003, TC_SECC_AC_VTB_ChargingStatus_004, TC_SECC_AC_VTB_ChargingStatus_006, TC_SECC_AC_VTB_ChargingStatus_008, TC_SECC_AC_VTB_PowerDelivery_001, TC_SECC_AC_VTB_PowerDelivery_002, TC_SECC_AC_VTB_PowerDelivery_003, TC_SECC_AC_VTB_PowerDelivery_004, TC_SECC_AC_VTB_PowerDelivery_005, TC_SECC_AC_VTB_PowerDelivery_006, TC_SECC_AC_VTB_PowerDelivery_007, TC_SECC_AC_VTB_PowerDelivery_008, TC_SECC_AC_VTB_PowerDelivery_009, TC_SECC_AC_VTB_PowerDelivery_010, TC_SECC_AC_VTB_PowerDelivery_011, TC_SECC_AC_VTB_PowerDelivery_014, TC_SECC_AC_VTB_ServiceDiscovery_001, TC_SECC_AC_VTB_ServiceDiscovery_002, TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_AC_VTB_SessionStop_001, TC_SECC_AC_VTB_SessionStop_002, TC_SECC_AC_VTB_SessionStop_003, TC_SECC_AC_VTB_ChargingStatus_002, TC_SECC_AC_VTB_ChargingStatus_007, TC_SECC_AC_VTB_ChargingStatus_009, TC_SECC_AC_VTB_ChargingStatus_010, TC_SECC_AC_VTB_ChargingStatus_011, TC_SECC_AC_VTB_ChargingStatus_012, TC_SECC_AC_VTB_MeteringReceipt_001, TC_SECC_AC_VTB_MeteringReceipt_002, TC_SECC_AC_VTB_MeteringReceipt_003, TC_SECC_AC_VTB_MeteringReceipt_004, TC_SECC_AC_VTB_MeteringReceipt_006, TC_SECC_AC_VTB_MeteringReceipt_007, TC_SECC_CMN_VTB_ChargeParameterDiscovery_001, TC_SECC_CMN_VTB_PaymentServiceSelection_001, TC_SECC_CMN_VTB_PaymentServiceSelection_004, TC_SECC_CMN_VTB_PaymentServiceSelection_005,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_PaymentServiceSelection_006, TC_SECC_CMN_VTB_PaymentServiceSelection_007, TC_SECC_CMN_VTB_ServiceDiscovery_001, TC_SECC_CMN_VTB_ServiceDiscovery_002, TC_SECC_CMN_VTB_ServiceDiscovery_003, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_CMN_VTB_SessionSetup_002, TC_SECC_CMN_VTB_Authorization_001, TC_SECC_CMN_VTB_Authorization_003, TC_SECC_CMN_VTB_Authorization_005, TC_SECC_CMN_VTB_Authorization_014, TC_SECC_CMN_VTB_PaymentServiceSelection_002, TC_SECC_CMN_VTB_PaymentServiceSelection_009, TC_SECC_CMN_VTB_ServiceDetail_001, TC_SECC_CMN_VTB_ServiceDetail_003, TC_SECC_CMN_VTB_ServiceDetail_005, TC_SECC_CMN_VTB_ServiceDetail_007, TC_SECC_CMN_VTB_ServiceDetail_011, TC_SECC_CMN_VTB_Authorization_002, TC_SECC_CMN_VTB_Authorization_004, TC_SECC_CMN_VTB_Authorization_006, TC_SECC_CMN_VTB_Authorization_007, TC_SECC_CMN_VTB_Authorization_010, TC_SECC_CMN_VTB_Authorization_011, TC_SECC_CMN_VTB_Authorization_012, TC_SECC_CMN_VTB_Authorization_013, TC_SECC_CMN_VTB_Authorization_015, TC_SECC_CMN_VTB_Authorization_016, TC_SECC_CMN_VTB_Authorization_017, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateInstallation_002, TC_SECC_CMN_VTB_CertificateInstallation_003, TC_SECC_CMN_VTB_CertificateInstallation_005, TC_SECC_CMN_VTB_CertificateInstallation_006, TC_SECC_CMN_VTB_CertificateInstallation_007, TC_SECC_CMN_VTB_CertificateInstallation_008, TC_SECC_CMN_VTB_CertificateUpdate_001, TC_SECC_CMN_VTB_CertificateUpdate_002, TC_SECC_CMN_VTB_CertificateUpdate_003, TC_SECC_CMN_VTB_CertificateUpdate_005, TC_SECC_CMN_VTB_CertificateUpdate_006, TC_SECC_CMN_VTB_CertificateUpdate_008, TC_SECC_CMN_VTB_CertificateUpdate_009, TC_SECC_CMN_VTB_CertificateUpdate_010, TC_SECC_CMN_VTB_CertificateUpdate_011, TC_SECC_CMN_VTB_CertificateUpdate_012, TC_SECC_CMN_VTB_CertificateUpdate_013, TC_SECC_CMN_VTB_CertificateUpdate_014, TC_SECC_CMN_VTB_CertificateUpdate_015, TC_SECC_CMN_VTB_CertificateUpdate_016, TC_SECC_CMN_VTB_CertificateUpdate_017, TC_SECC_CMN_VTB_PaymentDetails_001, TC_SECC_CMN_VTB_PaymentDetails_002, TC_SECC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_PaymentDetails_004, TC_SECC_CMN_VTB_PaymentDetails_005, TC_SECC_CMN_VTB_PaymentDetails_007,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_CMN_VTB_PaymentDetails_008, TC_SECC_CMN_VTB_PaymentDetails_009, TC_SECC_CMN_VTB_PaymentDetails_010, TC_SECC_CMN_VTB_PaymentServiceSelection_003, TC_SECC_CMN_VTB_PaymentServiceSelection_010, TC_SECC_CMN_VTB_ServiceDetail_002, TC_SECC_CMN_VTB_ServiceDetail_004, TC_SECC_CMN_VTB_ServiceDetail_006, TC_SECC_CMN_VTB_ServiceDetail_008, TC_SECC_CMN_VTB_ServiceDetail_012, TC_SECC_DC_VTB_CableCheck_001, TC_SECC_DC_VTB_CableCheck_002, TC_SECC_DC_VTB_CableCheck_003, TC_SECC_DC_VTB_CableCheck_004, TC_SECC_DC_VTB_CableCheck_006, TC_SECC_DC_VTB_CableCheck_007, TC_SECC_DC_VTB_CableCheck_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_002, TC_SECC_DC_VTB_ChargeParameterDiscovery_003, TC_SECC_DC_VTB_ChargeParameterDiscovery_004, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_CurrentDemand_001, TC_SECC_DC_VTB_CurrentDemand_003, TC_SECC_DC_VTB_CurrentDemand_004, TC_SECC_DC_VTB_CurrentDemand_006, TC_SECC_DC_VTB_CurrentDemand_008, TC_SECC_DC_VTB_PowerDelivery_001, TC_SECC_DC_VTB_PowerDelivery_002, TC_SECC_DC_VTB_PowerDelivery_003, TC_SECC_DC_VTB_PowerDelivery_004, TC_SECC_DC_VTB_PowerDelivery_005, TC_SECC_DC_VTB_PowerDelivery_006,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_PowerDelivery_007, TC_SECC_DC_VTB_PowerDelivery_008, TC_SECC_DC_VTB_PowerDelivery_011, TC_SECC_DC_VTB_PowerDelivery_012, TC_SECC_DC_VTB_PowerDelivery_013, TC_SECC_DC_VTB_PowerDelivery_014, TC_SECC_DC_VTB_PowerDelivery_015, TC_SECC_DC_VTB_PowerDelivery_016, TC_SECC_DC_VTB_PowerDelivery_017, TC_SECC_DC_VTB_PowerDelivery_018, TC_SECC_DC_VTB_PowerDelivery_019, TC_SECC_DC_VTB_PowerDelivery_020, TC_SECC_DC_VTB_PowerDelivery_021, TC_SECC_DC_VTB_PreCharge_001, TC_SECC_DC_VTB_PreCharge_002, TC_SECC_DC_VTB_PreCharge_003, TC_SECC_DC_VTB_ServiceDiscovery_001, TC_SECC_DC_VTB_ServiceDiscovery_002, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006, TC_SECC_DC_VTB_SessionStop_001, TC_SECC_DC_VTB_SessionStop_002, TC_SECC_DC_VTB_SessionStop_003, TC_SECC_DC_VTB_SessionStop_004, TC_SECC_DC_VTB_SessionStop_006, TC_SECC_DC_VTB_SessionStop_007, TC_SECC_DC_VTB_SessionStop_008, TC_SECC_DC_VTB_WeldingDetection_001, TC_SECC_DC_VTB_WeldingDetection_002, TC_SECC_DC_VTB_WeldingDetection_003, TC_SECC_DC_VTB_WeldingDetection_005, TC_SECC_DC_VTB_CurrentDemand_002, TC_SECC_DC_VTB_CurrentDemand_007, TC_SECC_DC_VTB_CurrentDemand_009, TC_SECC_DC_VTB_CurrentDemand_010, TC_SECC_DC_VTB_CurrentDemand_011, TC_SECC_DC_VTB_CurrentDemand_012, TC_SECC_DC_VTB_MeteringReceipt_001, TC_SECC_DC_VTB_MeteringReceipt_002, TC_SECC_DC_VTB_MeteringReceipt_003, TC_SECC_DC_VTB_MeteringReceipt_004, TC_SECC_DC_VTB_MeteringReceipt_006, TC_SECC_DC_VTB_MeteringReceipt_007,
Group_[V2G2-659]_[V2G2-660]_[V2G2-661]_[V2G2-662]	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001, TC_EVCC_AC_VTB_ChargeParameterDiscovery_008, TC_EVCC_AC_VTB_ChargeParameterDiscovery_011, TC_EVCC_AC_VTB_ChargeParameterDiscovery_012, TC_EVCC_AC_VTB_ChargeParameterDiscovery_013, TC_EVCC_AC_VTB_ChargeParameterDiscovery_014, TC_EVCC_AC_VTB_ChargeParameterDiscovery_015, TC_EVCC_AC_VTB_ChargeParameterDiscovery_016, TC_EVCC_AC_VTB_ChargeParameterDiscovery_017, TC_EVCC_AC_VTB_ChargeParameterDiscovery_018, TC_EVCC_AC_VTB_ChargeParameterDiscovery_019, TC_EVCC_AC_VTB_ChargeParameterDiscovery_020, TC_EVCC_AC_VTB_ChargeParameterDiscovery_021, TC_EVCC_AC_VTB_ChargeParameterDiscovery_022, TC_EVCC_AC_VTB_ChargeParameterDiscovery_023, TC_EVCC_AC_VTB_ChargeParameterDiscovery_024, TC_EVCC_AC_VTB_ChargeParameterDiscovery_025, TC_EVCC_AC_VTB_ChargeParameterDiscovery_026, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_ChargingStatus_001, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_SessionSetup_001, TC_EVCC_AC_VTB_SessionSetup_002, TC_EVCC_AC_VTB_SessionSetup_003, TC_EVCC_AC_VTB_SessionSetup_004, TC_EVCC_AC_VTB_SessionStop_001, TC_EVCC_AC_VTB_SessionStop_009, TC_EVCC_AC_VTB_SessionStop_012, TC_EVCC_AC_VTB_SessionStop_015, TC_EVCC_AC_VTB_SessionStop_018, TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_AC_VTB_SessionStop_007, TC_EVCC_AC_VTB_SessionStop_008, TC_EVCC_AC_VTB_SessionStop_010, TC_EVCC_AC_VTB_SessionStop_011, TC_EVCC_AC_VTB_SessionStop_013, TC_EVCC_AC_VTB_SessionStop_014, TC_EVCC_AC_VTB_SessionStop_016, TC_EVCC_AC_VTB_SessionStop_017, TC_EVCC_AC_VTB_SessionStop_019, TC_EVCC_AC_VTB_SessionStop_020, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_CMN_VTB_ServiceDiscovery_001, TC_EVCC_CMN_VTB_SessionSetup_001, TC_EVCC_CMN_VTB_SessionSetup_008, TC_EVCC_CMN_VTB_SupportedAppProtocol_001, TC_EVCC_CMN_VTB_Authorization_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_CMN_VTB_ServiceDetail_002, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_CMN_VTB_SupportedAppProtocol_004, TC_EVCC_CMN_VTB_SupportedAppProtocol_005, TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_CMN_VTB_ServiceDetail_001, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_CableCheck_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_001, TC_EVCC_DC_VTB_ChargeParameterDiscovery_008, TC_EVCC_DC_VTB_ChargeParameterDiscovery_011, TC_EVCC_DC_VTB_ChargeParameterDiscovery_012, TC_EVCC_DC_VTB_ChargeParameterDiscovery_013, TC_EVCC_DC_VTB_ChargeParameterDiscovery_014, TC_EVCC_DC_VTB_ChargeParameterDiscovery_015, TC_EVCC_DC_VTB_ChargeParameterDiscovery_016, TC_EVCC_DC_VTB_ChargeParameterDiscovery_017, TC_EVCC_DC_VTB_ChargeParameterDiscovery_018, TC_EVCC_DC_VTB_ChargeParameterDiscovery_019, TC_EVCC_DC_VTB_ChargeParameterDiscovery_020, TC_EVCC_DC_VTB_ChargeParameterDiscovery_021, TC_EVCC_DC_VTB_ChargeParameterDiscovery_022, TC_EVCC_DC_VTB_ChargeParameterDiscovery_023, TC_EVCC_DC_VTB_ChargeParameterDiscovery_024, TC_EVCC_DC_VTB_ChargeParameterDiscovery_025, TC_EVCC_DC_VTB_ChargeParameterDiscovery_026, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_CurrentDemand_001, TC_EVCC_DC_VTB_PreCharge_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_SessionSetup_001, TC_EVCC_DC_VTB_SessionSetup_002, TC_EVCC_DC_VTB_SessionSetup_003, TC_EVCC_DC_VTB_SessionSetup_004, TC_EVCC_DC_VTB_SessionStop_001, TC_EVCC_DC_VTB_SessionStop_002, TC_EVCC_DC_VTB_SessionStop_007, TC_EVCC_DC_VTB_SessionStop_008, TC_EVCC_DC_VTB_SessionStop_013, TC_EVCC_DC_VTB_SessionStop_014, TC_EVCC_DC_VTB_SessionStop_019, TC_EVCC_DC_VTB_SessionStop_020, TC_EVCC_DC_VTB_SessionStop_025, TC_EVCC_DC_VTB_SessionStop_026, TC_EVCC_DC_VTB_WeldingDetection_001, TC_EVCC_DC_VTB_WeldingDetection_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_SessionStop_003, TC_EVCC_DC_VTB_SessionStop_004, TC_EVCC_DC_VTB_SessionStop_005, TC_EVCC_DC_VTB_SessionStop_006, TC_EVCC_DC_VTB_SessionStop_009, TC_EVCC_DC_VTB_SessionStop_010, TC_EVCC_DC_VTB_SessionStop_011, TC_EVCC_DC_VTB_SessionStop_012, TC_EVCC_DC_VTB_SessionStop_015, TC_EVCC_DC_VTB_SessionStop_016, TC_EVCC_DC_VTB_SessionStop_017, TC_EVCC_DC_VTB_SessionStop_018, TC_EVCC_DC_VTB_SessionStop_021, TC_EVCC_DC_VTB_SessionStop_022, TC_EVCC_DC_VTB_SessionStop_023, TC_EVCC_DC_VTB_SessionStop_024, TC_EVCC_DC_VTB_SessionStop_027, TC_EVCC_DC_VTB_SessionStop_028, TC_EVCC_DC_VTB_SessionStop_029, TC_EVCC_DC_VTB_SessionStop_030, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007,
Group_[V2G2-434]_[V2G2-436]_[V2G2-437]_[V2G2-438]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007, TC_EVCC_AC_VTB_ChargingStatus_005, TC_EVCC_AC_VTB_PowerDelivery_009, TC_EVCC_AC_VTB_SessionStop_005, TC_EVCC_AC_VTB_MeteringReceipt_007, TC_EVCC_CMN_VTB_Authorization_008, TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009, TC_EVCC_CMN_VTB_ServiceDiscovery_005, TC_EVCC_CMN_VTB_SessionSetup_004, TC_EVCC_CMN_VTB_SupportedAppProtocol_003, TC_EVCC_CMN_VTB_ServiceDetail_011, TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005, TC_EVCC_CMN_VTB_CertificateInstallation_009, TC_EVCC_CMN_VTB_CertificateUpdate_011, TC_EVCC_CMN_VTB_ServiceDetail_012, TC_EVCC_DC_VTB_CableCheck_005, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007, TC_EVCC_DC_VTB_CurrentDemand_005, TC_EVCC_DC_VTB_PreCharge_005, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009, TC_EVCC_DC_VTB_WeldingDetection_005, TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005, TC_EVCC_DC_VTB_MeteringReceipt_007,
Group_[V2G2-190]_[V2G2-191]_[V2G2-543]_[V2G2-192]	TC_SECC_AC_VTB_SessionSetup_001, TC_SECC_AC_VTB_SessionSetup_002, TC_SECC_AC_VTB_SessionSetup_003, TC_SECC_AC_VTB_SessionSetup_004, TC_SECC_AC_VTB_SessionSetup_005, TC_SECC_AC_VTB_SessionSetup_006, TC_SECC_CMN_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_001, TC_SECC_DC_VTB_SessionSetup_002, TC_SECC_DC_VTB_SessionSetup_003, TC_SECC_DC_VTB_SessionSetup_004, TC_SECC_DC_VTB_SessionSetup_005, TC_SECC_DC_VTB_SessionSetup_006,
Group_[V2G2-221]_[V2G2-222]_[V2G2-286]_[V2G2-510]	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009, TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009, TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014,
Group_[V2G2-836]_[V2G2-284]_[V2G2-606]_[V2G2-288]_[V2G2-607]	TC_EVCC_AC_VTB_PowerDelivery_001, TC_EVCC_AC_VTB_PowerDelivery_013, TC_EVCC_AC_VTB_PowerDelivery_014, TC_EVCC_AC_VTB_PowerDelivery_015, TC_EVCC_AC_VTB_PowerDelivery_016, TC_EVCC_AC_VTB_PowerDelivery_017, TC_EVCC_AC_VTB_PowerDelivery_018, TC_EVCC_AC_VTB_PowerDelivery_019, TC_EVCC_AC_VTB_PowerDelivery_020, TC_EVCC_AC_VTB_PowerDelivery_021, TC_EVCC_AC_VTB_PowerDelivery_022, TC_EVCC_AC_VTB_PowerDelivery_023, TC_EVCC_AC_VTB_PowerDelivery_024, TC_EVCC_AC_VTB_PowerDelivery_025, TC_EVCC_AC_VTB_PowerDelivery_026, TC_EVCC_AC_VTB_PowerDelivery_027, TC_EVCC_AC_VTB_PowerDelivery_028, TC_EVCC_AC_VTB_PowerDelivery_029, TC_EVCC_AC_VTB_PowerDelivery_030, TC_EVCC_AC_VTB_PowerDelivery_031, TC_EVCC_AC_VTB_PowerDelivery_032, TC_EVCC_AC_VTB_PowerDelivery_033, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021, TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022,
Group_[V2G2-294]_[V2G2-608]_[V2G2-299]_[V2G2-609]_[V2G2-300]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_002, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_023, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_025, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_028, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_005, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_023, TC_SECC_DC_VTB_ChargeParameterDiscovery_024, TC_SECC_DC_VTB_ChargeParameterDiscovery_025, TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_028, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
Group_[V2G2-201]_[V2G2-202]_[V2G2-431]_[V2G2-349]_[V2G2-350]_[V2G2-351]_[V2G2-352]	TC_EVCC_AC_VTB_PaymentServiceSelection_001, TC_EVCC_AC_VTB_PaymentServiceSelection_003, TC_EVCC_AC_VTB_PaymentServiceSelection_005, TC_EVCC_AC_VTB_PaymentServiceSelection_007, TC_EVCC_AC_VTB_PaymentServiceSelection_009, TC_EVCC_AC_VTB_PaymentServiceSelection_011, TC_EVCC_AC_VTB_PaymentServiceSelection_013, TC_EVCC_AC_VTB_PaymentServiceSelection_015, TC_EVCC_AC_VTB_PaymentServiceSelection_017, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_002, TC_EVCC_AC_VTB_PaymentServiceSelection_004, TC_EVCC_AC_VTB_PaymentServiceSelection_006, TC_EVCC_AC_VTB_PaymentServiceSelection_008, TC_EVCC_AC_VTB_PaymentServiceSelection_010, TC_EVCC_AC_VTB_PaymentServiceSelection_012, TC_EVCC_AC_VTB_PaymentServiceSelection_014, TC_EVCC_AC_VTB_PaymentServiceSelection_016, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007, TC_EVCC_CMN_VTB_PaymentServiceSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_004, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_CMN_VTB_PaymentServiceSelection_001, TC_EVCC_CMN_VTB_PaymentServiceSelection_003, TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_002, TC_EVCC_DC_VTB_PaymentServiceSelection_004, TC_EVCC_DC_VTB_PaymentServiceSelection_006, TC_EVCC_DC_VTB_PaymentServiceSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_010, TC_EVCC_DC_VTB_PaymentServiceSelection_012, TC_EVCC_DC_VTB_PaymentServiceSelection_014, TC_EVCC_DC_VTB_PaymentServiceSelection_016, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008, TC_EVCC_DC_VTB_PaymentServiceSelection_001, TC_EVCC_DC_VTB_PaymentServiceSelection_003, TC_EVCC_DC_VTB_PaymentServiceSelection_005, TC_EVCC_DC_VTB_PaymentServiceSelection_007, TC_EVCC_DC_VTB_PaymentServiceSelection_009, TC_EVCC_DC_VTB_PaymentServiceSelection_011, TC_EVCC_DC_VTB_PaymentServiceSelection_013, TC_EVCC_DC_VTB_PaymentServiceSelection_015, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005, TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
Group_[V2G2-353]_[V2G2-354]_[V2G2-004]_[V2G2-009]_[V2G2-656]_[V2G2-926]_[V2G2-822]	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_CMN_VTB_PaymentDetails_001, TC_EVCC_CMN_VTB_PaymentDetails_002, TC_EVCC_CMN_VTB_PaymentDetails_003, TC_SECC_CMN_VTB_CertificateInstallation_001, TC_SECC_CMN_VTB_CertificateUpdate_001,
Group_[V2G2-310]_[V2G2-313]_[V2G2-610]_[V2G2-611]_[V2G2-327]_[V2G2-614]_[V2G2-331]_[V2G2-672]	TC_SECC_AC_VTB_ChargeParameterDiscovery_001, TC_SECC_AC_VTB_ChargeParameterDiscovery_006, TC_SECC_AC_VTB_ChargeParameterDiscovery_007, TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_012, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_014, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_016, TC_SECC_AC_VTB_ChargeParameterDiscovery_017, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_019, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_022, TC_SECC_AC_VTB_ChargeParameterDiscovery_029, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_031, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_033, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_AC_VTB_ChargeParameterDiscovery_036, TC_SECC_DC_VTB_ChargeParameterDiscovery_001, TC_SECC_DC_VTB_ChargeParameterDiscovery_006, TC_SECC_DC_VTB_ChargeParameterDiscovery_007, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_012, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_014, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_016, TC_SECC_DC_VTB_ChargeParameterDiscovery_017, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_019, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_022, TC_SECC_DC_VTB_ChargeParameterDiscovery_029, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_031, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_033, TC_SECC_DC_VTB_ChargeParameterDiscovery_034,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_036,
Group_[V2G2-117]_[V2G2-119]_[V2G2-764]_[V2G2-765]_[V2G2-766]_[V2G2-767]_[V2G2-768]_[V2G2-769]_[V2G2-770]_[V2G2-771]	TC_EVCC_AC_VTB_MeteringReceipt_001, TC_EVCC_AC_VTB_MeteringReceipt_008, TC_EVCC_AC_VTB_MeteringReceipt_009, TC_EVCC_AC_VTB_MeteringReceipt_013, TC_EVCC_AC_VTB_MeteringReceipt_014, TC_EVCC_AC_VTB_MeteringReceipt_015, TC_EVCC_AC_VTB_MeteringReceipt_016, TC_EVCC_AC_VTB_MeteringReceipt_017, TC_EVCC_AC_VTB_MeteringReceipt_018, TC_EVCC_AC_VTB_MeteringReceipt_019, TC_EVCC_AC_VTB_MeteringReceipt_020, TC_EVCC_AC_VTB_MeteringReceipt_021, TC_EVCC_CMN_VTB_Authorization_001, TC_EVCC_CMN_VTB_Authorization_010, TC_EVCC_CMN_VTB_Authorization_011, TC_EVCC_CMN_VTB_CertificateInstallation_001, TC_EVCC_CMN_VTB_CertificateUpdate_001, TC_EVCC_DC_VTB_MeteringReceipt_001, TC_EVCC_DC_VTB_MeteringReceipt_008, TC_EVCC_DC_VTB_MeteringReceipt_009, TC_EVCC_DC_VTB_MeteringReceipt_013, TC_EVCC_DC_VTB_MeteringReceipt_014, TC_EVCC_DC_VTB_MeteringReceipt_015, TC_EVCC_DC_VTB_MeteringReceipt_016, TC_EVCC_DC_VTB_MeteringReceipt_017, TC_EVCC_DC_VTB_MeteringReceipt_018, TC_EVCC_DC_VTB_MeteringReceipt_019, TC_EVCC_DC_VTB_MeteringReceipt_020, TC_EVCC_DC_VTB_MeteringReceipt_021,
Group_([V2G2-307]_[V2G2-308]_pnC_[V2G2-309]_pnC_[V2G2-905]_pnC_([V2G2-906]_[V2G2-316]_[V2G2-612]_[V2G2-321]_[V2G2-613]	TC_SECC_AC_VTB_ChargeParameterDiscovery_008, TC_SECC_AC_VTB_ChargeParameterDiscovery_009, TC_SECC_AC_VTB_ChargeParameterDiscovery_010, TC_SECC_AC_VTB_ChargeParameterDiscovery_011, TC_SECC_AC_VTB_ChargeParameterDiscovery_013, TC_SECC_AC_VTB_ChargeParameterDiscovery_015, TC_SECC_AC_VTB_ChargeParameterDiscovery_018, TC_SECC_AC_VTB_ChargeParameterDiscovery_020, TC_SECC_AC_VTB_ChargeParameterDiscovery_021, TC_SECC_AC_VTB_ChargeParameterDiscovery_024, TC_SECC_AC_VTB_ChargeParameterDiscovery_026, TC_SECC_AC_VTB_ChargeParameterDiscovery_027, TC_SECC_AC_VTB_ChargeParameterDiscovery_030, TC_SECC_AC_VTB_ChargeParameterDiscovery_032, TC_SECC_AC_VTB_ChargeParameterDiscovery_034, TC_SECC_AC_VTB_ChargeParameterDiscovery_035, TC_SECC_DC_VTB_ChargeParameterDiscovery_008, TC_SECC_DC_VTB_ChargeParameterDiscovery_009, TC_SECC_DC_VTB_ChargeParameterDiscovery_010, TC_SECC_DC_VTB_ChargeParameterDiscovery_011, TC_SECC_DC_VTB_ChargeParameterDiscovery_013, TC_SECC_DC_VTB_ChargeParameterDiscovery_015, TC_SECC_DC_VTB_ChargeParameterDiscovery_018, TC_SECC_DC_VTB_ChargeParameterDiscovery_020, TC_SECC_DC_VTB_ChargeParameterDiscovery_021, TC_SECC_DC_VTB_ChargeParameterDiscovery_024,

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Group name	TC IDs
	TC_SECC_DC_VTB_ChargeParameterDiscovery_026, TC_SECC_DC_VTB_ChargeParameterDiscovery_027, TC_SECC_DC_VTB_ChargeParameterDiscovery_030, TC_SECC_DC_VTB_ChargeParameterDiscovery_032, TC_SECC_DC_VTB_ChargeParameterDiscovery_034, TC_SECC_DC_VTB_ChargeParameterDiscovery_035,

7.6 Test case description

The test case descriptions in this document are described according to the template shown in Table 27.

Table 27 — Test case description template

TC Id	The TC Id is a unique identifier for a test case. It is specified according to the TC Id naming convention specified in 7.4.3.
Test objective	Short description of test objective according to the requirements from ISO 15118-2.
Document reference	The document reference indicates the sub-clauses in the referenced standard where the static or dynamic conformance requirement(s) is/are specified. The references are provided according to the following format: Document: ISO15118-X:20XX(IS|FDIS) Section(s): x.x.x.x.x, y.y.y.y.y, …
Referenced requirement(s)	List of referenced requirement(s) for this test case. The requirements are referenced according to the format specified in the referenced standard, for example: [V2GX-XXX]
Config Id	The Config Id references the test configuration specified in this document according to 7.3.2 which is selected for this test case. For example: CF_04_001
PICS selection	The PICS selection references the PICS statement(s) for this test case in accordance with 7.3.4. if the corresponding test behavior requires a specific assignment. Example: `PICS_CMN_CMN_IdentificationMode := eIM`
PIXIT selection	The PIXIT selection references the PIXIT statement(s) for this test case in accordance with 7.4.7. if the corresponding test behavior requires a specific assignment on SUT side. Example: `PIXIT_EVCC_CMN_VAS := serviceDetail`
Pre-condition	
The pre-condition specifies in which state the SUT is before executing the actual test case. In the corresponding test case, when the execution of the initial condition does not succeed, it leads to the assignment of an “Inconclusive” verdict. The pre-condition is specified as reference to the pre-condition function in the ATS. Example: `f_SECC_CMN_PR_ServiceDetail_001`	

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Expected behavior
Definition of the events, which are part of the test case objective, the SUT is expected to perform in order to conform to the base specification. The expected behavior is specified as reference to the testbehavior function in the ATS. Example: `f_SECC_CMN_TB_VTB_ServiceDetail_001`

7.7 Test case specification

7.7.1 General information

The abstract test case specifications are provided in the form of a TTCN-3 test suite according to the TTCN-3 Core Language as specified in ETSI ES 201 873-1 V4.6.1. This sub clause specifies additional conventions for the scope of this test suite

7.7.2 Data types

7.7.2.1 V2G PICS

The TTCN-3 data structures for PICS (enumerations) according to 7.3.4 are specified in Annex P.2.

7.7.2.2 V2G TCP Port Control

The TTCN-3 data structures for V2G TCP Port Control according to [V2G4-007] are specified in Annex P.4.

7.7.2.3 V2GTP

The TTCN-3 data structures for testing V2GTP conformance are specified in Annex P.5. The TTCN-3 data representation for V2GTP message header handling is specified according to ISO 15118-2:2014, 7.8.

7.7.2.4 SDP

The TTCN-3 data structures for testing SDP conformance are specified in Annex P.5. The TTCN-3 data representation for SDP requests and responses is specified according to ISO 15118-2:2014, 7.10.1.

7.7.2.5 V2G messages

The TTCN-3 data representation of all V2G messages specified in ISO 15118-2:2014, Clause 8 is handled according to ETSI ES 201 873-9 V4.6.1 (TTCN-3: Using XML schema with TTCN-3). It specifies an unambiguous mapping between XML Schema based type definitions and a corresponding TTCN-3 data type representation. The resulting TTCN-3 data structures for testing V2G messaging conformance are specified in Annex P.7.

7.7.2.6 X.509 Certificates

The TTCN-3 data representation of X.509 certificates is specified in addition to ISO 15118-2:2014, Clause 8 in XML schema and is handled according to ETSI ES 201 873-9 V4.6.1 (TTCN-3: Using XML schema with TTCN-3). It specifies an unambiguous mapping between XML Schema based type definitions and a corresponding TTCN-3 data type representation. The resulting TTCN-3 data structures for testing V2G messaging conformance including X.509 Certificates are specified in Annex P.7.

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

7.7.3 Templates

A template specifies at least one value for a data type. The templates in this document are used for two purposes:

— Representation of stimulus test data

— SUT response matching with expected behavior

The templates relevant for this document are specified in Annex K - Annex O. Templates that are only used once in a concrete test case are specified in their corresponding context in the Annexes. All templates that are used in several test cases / contexts are specified in the common sections in the Annexes.

The templates for the V2G messages specified in the XML schema of ISO 15118-2 consist of a header (`MessageHeaderType`) and a body (`BodyElement_group`). There are templates that can be parameterized. At least one template for each message exists, where all elements of the first level of the message type are specified as parameters of the template.

In the TSS the following rules apply for templates used in the context of test cases for EVCC / SECC:

— For testing an SECC every attribute in an EVCC request message is either allocated or omitted.

— For testing an EVCC every attribute in an SECC response message is allocated either by parameters of the template or TTCN-3 wildcards.

7.7.4 Timeouts and timers

All timeouts are specified according to ISO 15118-2. For the scope of this document *performance times* in ISO 15118-2 are irrelevant since they are not visible to the test system (black box testing). Hence only *timeouts* can be exactly measured from the test system''s perspective and are considered in this document. The resulting timeout definitions for the TTCN-3 test suite are provided in Annex A.2.

The timer handling in the test suite is specified as follows: Before sending a stimulus to the SUT the corresponding message timer is started. After receiving the expected response from the SUT the message timer will be stopped. In case no response is received from the SUT until the point in time specified by the corresponding timeout value, the test verdict is set t‘ 'fal’e'. The timer handling is modelled as part of the test behavior section within the abstract test case specification.

7.7.5 Common behavior

Within the TSS few functions are specified which are used in different contexts thus being invoked from various callers. Such functions are specified in Annex F.5.

7.7.6 Test case modelling

One test case always executes a specific test behavior and verifies the SUT behavior according to its message-based response. All test cases always begin and end in a specified and recoverable state of the SUT and the test system. As a consequence, every test case consists of the following phases:

a) Initialize

   1) Create PTC components

   2) Map all MTC/PTC/system ports and communication bindings according to the underlying test configuration

b) Pre-condition

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

The test system is supposed to start in a state so that a communication or data exchange with the SUT is possible. In the pre-condition of a test case the test system as well as the SUT therefore initialize into known and stable state. This incorporates the following action:

1) Bring SUT and test system to a valid and known state before the actual test behavior is executed

c) Test behavior

The actual test behavior specifies the set of steps during the test execution which are essential in order to achieve the test purpose and assign verdicts to the possible outcomes. This involves the following actions:

1) Initialize and start relevant timer(s) on the test system side

2) Execute test behavior (send stimulus to the SUT)

3) Listen to any events and verify SUT response (stop timers)

4) Assign test verdict for this test behavior

d) Post-condition

1) Bring SUT and test system into final/initial state

e) Shutdown

1) Unmap all connected ports & communication bindings

2) Shutdown PTC components

7.7.7 Message handling depending on SUT type

In reference to the test case modelling as specified in the previous sub clause, a stimulus is always sent from the test system to the SUT as part of the test behavior (phase c2). Due to the strictly specified client, server roles assigned to the EVCC and SECC respectively, the following two message handling paradigms result depending on the type of the SUT:

[V2G4-067] If the SUT is an SECC, the test system shall send a request message as stimulus to the SUT during phase c2 (see 7.7.6) and expects the corresponding V2G response message back from the SUT for verdict assignment.

[V2G4-068] If the SUT is an EVCC, the test system shall send a response message as stimulus to the SUT during phase c2 (see 7.7.6) and expects a V2G request message according to the next message pattern in the V2G message sequence back from the SUT for verdict assignment. If multiple next messages are specified, the test system shall be able to handle these for verdict assignment.

7.7.8 Configuration of electrical parameters and power flow related data

7.7.8.1 General information

Some test cases in this document depend on electrical parameterization of AC or DC power flow related data. The parameterization recommendations provided in sub clauses 7.7.8.2 and 7.7.8.3 ensure that general test conditions are satisfied by applying values that are in the supported range of the test system as well as the SUT.

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

7.7.8.2 Electrical parameters for DC power flow configuration

With reference to Annex A.2, an overview of all relevant DC power flow test system parameters and their recommended settings are provided in Table 28, where the following definitions apply:

- Test System MaxDCxxxRating – Maximum nominal DC rating of the Test System for current, voltage and power
- Test System MinDCxxxRating – Minimum nominal DC rating of the Test System for current and voltage.
- SUT MaxDCxxxRating – Maximum nominal DC rating of the SUT for current, voltage and power
- SUT MinDCxxxRating – Minimum nominal DC rating of the SUT for current and voltage

Table 28 – DC power flow test system configuration parameters

Parameter	Description & recommended setting
`par_EVMaximumCurrentLimit (DC)`	Maximum DC current limit the test system applies in test configuration CF_04_001 (SUT SECC) Recommended setting: Min(Test System MaxDCCurrentRating, SUT MaxDCCurrentRating), where Test System MaxDCCurrentRating > SUT MinDCCurrentRating
`par_EVMaximumVoltageLimit (DC)`	Maximum DC voltage limit the test system applies in test configuration CF_04_001 (SUT SECC) Recommended setting: Min(Test System MaxDCVoltageRating, SUT MaxDCVoltageRating), where Test System MaxDCVoltageRating > SUT MinDCVoltageRating
`par_EVMaximumPowerLimit (DC)`	Maximum DC power limit the test system applies in test configuration CF_04_001 (SUT SECC) Recommended setting: Min(Test System MaxDCPowerRating, SUT MaxPowerRating)
`par_EVMinimumCurrentLimit (DC)`	Minimum DC current limit the test system applies in test configuration CF_04_001 (SUT SECC) Recommended setting: Max(Test System MinDCCurrentRating, SUT MinDCCurrentRating), where Test System MinDCCurrentRating < SUT MaxDCCurrentRating
`par_EVMinimumVoltageLimit (DC)`	Minimum DC voltage limit the test system applies in test configuration CF_04_001 (SUT SECC) Recommended setting: Max(Test System MinDCVoltageRating, SUT MinDCVoltageRating), where Test System MinDCVoltageRating < SUT MaxDCVoltageRating
`par_EVSEMaximumCurrentLimit (DC)`	Maximum DC current limit the test system applies in test configuration CF_04_002 (SUT EVCC) Recommended setting:

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

	Min(Test System MaxDCCurrentRating, SUT MaxDCCurrentRating), where Test System MaxDCCurrentRating > SUT MinDCCurrentRating
`par_EVSEMaximumVoltageLimit (DC)`	Maximum DC voltage limit the test system applies in test configuration CF_04_002 (SUT EVCC) Recommended setting: Min(Test System MaxDCVoltageRating, SUT MaxDCVoltageRating), where Test System MaxDCVoltageRating > SUT MinDCVoltageRating
`par_EVSEMaximumPowerLimit (DC)`	Maximum DC power limit the test system applies in test configuration CF_04_002 (SUT EVCC) Recommended setting: Min(Test System MaxDCPowerRating, SUT MaxPowerRating)
`par_EVSEMinimumCurrentLimit (DC)`	Minimum DC current limit the test system applies in test configuration CF_04_002 (SUT EVCC) Recommended setting: Max(Test System MinDCCurrentRating, SUT MinDCCurrentRating) , where Test System MinDCCurrentRating < SUT MaxDCCurrentRating
`par_EVSEMinimumVoltageLimit (DC)`	Minimum DC voltage limit the test system applies in test configuration CF_04_002 (SUT EVCC) Recommended setting: Max(Test System MinDCVoltageRating, SUT MinDCVoltageRating), where Test System MinDCVoltageRating < SUT MaxDCVoltageRating

NOTE 1 The parameterization recommendation is provided to ensure valid DC power flow related configuration for the purpose of charging communication protocol conformance test cases specified in this document. For validation of electrical requirements of the DC charging process, please refer to IEC 61851-1 and IEC 61851-23.

7.7.8.3 Electrical parameters for AC power flow configuration

With reference to Annex A.2, an overview of all relevant AC power flow test system parameters an their recommended settings are provided in Table 29, where the following definitions apply:

- Test System MaxACxxxRating – Maximum nominal AC rating of the Test System for current and voltage
- Test System MinACxxxRating – Minimum nominal AC current rating of the Test System
- SUT MaxACxxxRating – Maximum nominal AC rating of the SUT for current and voltage
- SUT MinACxxxRating – Minimum nominal AC rating of the SUT for current and voltage

Table 29 – AC power flow test system configuration parameters

Parameter	Description & recommended setting
par_EVMaxCurrent (AC)	Maximum AC current the test system applies in test configuration CF_04_001 (SUT SECC) Recommended setting:

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

	Min(Test System MaxACCurrentRating, SUT MaxACCurrentRating), where Test System MaxACCurrentRating > SUT MinACCurrentRating
par_EVMinCurrent (AC)	Minimum AC current the test system applies in test configuration CF_04_001 (SUT SECC) Recommended setting: Max(Test System MinACCurrentRating, SUT MinACCurrentRating), where Test System MinACCurrentRating < SUT MaxACCurrentRating
par_EVMaxVoltage (AC)	Maximum AC voltage the test system applies in test configuration CF_04_001 (SUT SECC) Recommended setting: Min(Test System MaxACVoltageRating, SUT MaxACVoltageRating), where Test System MaxACVoltageRating > SUT MinACVoltageRating
par_EVSEMaxCurrent (AC)	Maximum AC current the test system applies in test configuration CF_04_002 (SUT EVCC) Recommended setting: Min(Test System MaxACCurrentRating, SUT MaxACCurrentRating) where Test System MaxACCurrentRating > SUT MinACCurrentRating
par_EVSENominalVoltage (AC)	Nominal AC voltage the test system applies in test configuration CF_04_002 (SUT EVCC) Recommended setting: < SUT MaxACVoltageRating

NOTE 1 The parameterization recommendation is provided to ensure valid AC power flow related configuration for the purpose of test cases on communication protocol conformance specified in this document. For validation of electrical requirements of the AC charging process, please refer to IEC 61851-1.

7.7.9 Public key infrastructure (PKI) setup and certificate handling

The ISO 15118-2 requires the existence of multiple certificates to be applied for both the transport layer security (TLS) and for XML-security on application layer. SECC certificates are used to unilaterally authenticate SECC's by EVCC'S and contract certificates are handled at the application layer to authenticate an EVCC against an SECC and/or secondary actor. V2G root certificates and possibly Sub-CA certificates are used to certify SECC certificates and EVCC contract certificates. In addition, OEM root certificates and OEM provisioning certificates can be used for installing and updating contract certificates.

Figure 4 provides a schematic of the public key infrastructure (PKI) setup used for testing of ISO 15118-2. To verify the SUT behavior not only with valid credentials, more than one PKI is required which is indicated in Figure 4 by *#V2G-CAs, #MO-CAs, #OEM-CAs and #PE-CAs*. Figure 4 also provides the number of Sub-CAs and certificates (#CPO-Sub-CA, #Prov-Sub-CA, #MO-Sub-CA, #OEM-Sub-CA etc.) for every certificate chain. Not all permutations of these parameter are valid in terms of ISO 15118-2 requirements and are even not necessary at all. For reproducibility, Table 33 to Table 35 specify the minimal PKI configuration required for ISO 15118-2 testing. Since PKI-Tooling and the PKI-Timing depending on the overall test architecture, a specific setup or even a specific certificate set are not part of standardization.

These credentials are handled as PICS/PIXIT. Each certificate also implies the existence of the associated private and public key as well as a certificate chain if required. Table 30, Table 31 and Table 32 specify the test suite naming convention in this document for the PKI and certificate identifiers. Furthermore, some test case will also modify these PICS/PIXIT in order to verify the SUT behavior using *manipulated credentials*. The PICS/

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

PIXITs are a prerequisite before test case execution can be performed. The process to provide these Information to the SUT is out of the scope of this document.

Figure 4 — Schematic public key infrastructure setup for testing of ISO 15118-2

Table 30 — ISO 15118 test suite naming convention for PKI identifiers

Identifier	Values	Description
`<prefix>-<root_ca_nn>`	`PKI-{xxx}`	The prefix for the PKI identifiers Number of the root CA certificate (*#V2G-Cas*) NOTE Every V2G root CA specifies a new instance of the PKI.

Table 31 — ISO 15118 test suite naming convention for certificate identifiers

Identifier	Values	Description
`<pki_name>`	`{pki_name}`	See Table 30
`<prefix>`	`CRT`	The prefix for certificate identifiers
`<cert_dom>`		Domain the certificate belongs to

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

	V2G	If the certificate is issued for a V2G PKI service
	CPO	If the certificate is issued for a charge point operator
	PROV	If the certificate is issued for a provisioning service
	MO	If the certificate is issued for a mobility aperator
	OEM	If the certificate is issued for an OEM
	PE	If the certificate is issued for a private environment
	EVSE	If the certificate is issued for a EVSE by a CPO
	CONTRACT	If the certificate is issued for a contract by MO
<cert_hierarchy>		Hierarchy level of the certificate
	ROOT	If the certificate is a root CA (CA == true)
	SUB1	If the certificate is a CA (CA == true) and (PathLength == 1)
	SUB2	If the certificate is a CA (CA == true) and (PathLength == 0)
	LEAF	If the certificate is a LEAF (CA == false)
<cert_type>		Type of the certificate
	VALID	If the certificate is valid and the time of expire will not influent the test process
	EXPIRED	If the certificate is expired before test case execution
	EXPIRES	If the certificate expires soon. The time limit is configured so that it will trigger the certificate update procedure.
	INVALID_CONTENT	If any critical content (see ISO 15118-2:2014 Annex F) is invalid
	INVALID_EMAID	If the emaid has an invalid syntax
	WRONG_EMAID	If the emaid differs from the original version when the contract was concluded (Syntax is correct)
	EXPIRED_ISSUER_SUB1	If the leaf certificate was derived from expired SUB 1 certificate
	EXPIRED_ISSUER_SUB2	If the leaf certificate was derived from expired SUB 2 certificate

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table 32 — ISO 15118 test suite naming convention for wildcards in certificate identifiers

Identifier	Values	Description
<wildcard>	*	The wildcard prefix for certificate identifiers is used in the PKI definition to abbreviate the pki- and certificate name until the <cert_type> in a given context.
<cert_type>		(See Table 37)

Table 33 — ISO 15118 parameters for Test PKI-1 (multiple root Cas)

PKI name	PKI-1	
PKI parameter name	**PKI Parameter Value**	**Certificate name**
#V2G-Cas / V2G-CA-ID	1 / 1	PKI-1_CRT_V2G_ROOT_VALID
#MO-Cas	1	PKI-1_CRT_MO_ROOT_VALID
#OEM-Cas	1	PKI-1_CRT_OEM_ROOT_VALID
#PE-Cas	0	-
#CPO-Sub-CA	2	PKI-1_CRT_CPO_SUB1_VALID, PKI-1_CRT_CPO_SUB2_VALID
#Prov-Sub-CA	2	PKI-1_CRT_PROV_SUB1_VALID, PKI-1_CRT_PROV_SUB2_VALID
#MO-Sub-CA	5	PKI-1_CRT_MO_SUB1_VALID, *_EXPIRED PKI-1_CRT_MO_SUB2_VALID *_EXPIRED, *_EXPIRED_ISSUER_SUB1
#OEM-Sub-CA	2	PKI-1_CRT_OEM_SUB1_VALID, PKI-1_CRT_OEM_SUB2_VALID
#CPO-EVSE-Leaf	2	PKI-1_CRT_EVSE_LEAF_VALID, *_ EXPIRED
#Prov-Leaf	2	PKI-1_CRT_PROV_LEAF_VALID, *_ EXPIRED
#MO-Contract-Leaf	8	PKI-1_CRT_CONTRACT_LEAF_VALID, *_ EXPIRED, * _ EXPIRES, *_ INVALID_CONTENT, *_INVALID_EMAID, *_ WRONG_EMAID, *_EXPIRED_ISSUER_SUB1, *_EXPIRED_ISSUER_SUB2
#OEM-Prov-Leaf	3	PKI-1_CRT_OEM_LEAF_VALID, *_ EXPIRED, *_INVALID_CONTENT
#PE_Leaf	0	-

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table 34 — ISO 15118 parameters for Test PKI-2 (single V2G root CA)

PKI name	PKI-2	
PKI parameter name	**PKI parameter value**	**Certificate name**
#V2G-Cas / V2G-CA-ID	1 / 2	PKI-2_CRT_V2G_ROOT_VALID
#MO-Cas	0	-
#OEM-Cas	0	-
#PE-Cas	0	-
#CPO-Sub-CA	2	PKI-2_CRT_CPO_SUB1_VALID, PKI-2_CRT_CPO_SUB2_VALID
#Prov-Sub-CA	2	PKI-2_CRT_PROV_SUB1_VALID, PKI-2_CRT_PROV_SUB2_VALID
#MO-Sub-CA	5	PKI-2_CRT_MO_SUB1_VALID, *_EXPIRED PKI-2_CRT_MO_SUB2_VALID *_EXPIRED, *_EXPIRED_ISSUER_SUB1
#OEM-Sub-CA	2	PKI-2_CRT_OEM_SUB1_VALID, PKI-2_CRT_OEM_SUB2_VALID
#CPO-EVSE-Leaf	2	PKI-2_CRT_EVSE_LEAF_VALID, *_ EXPIRED
#Prov-Leaf	2	PKI-2_CRT_PROV_LEAF_VALID, *_ EXPIRED
#MO-Contract-Leaf	8	PKI-2_CRT_CONTRACT_LEAF_VALID, *_ EXPIRED, * _ EXPIRES, *_ INVALID_CONTENT, *_INVALID_EMAID, *_ WRONG_EMAID, *_EXPIRED_ISSUER_SUB1, *_EXPIRED_ISSUER_SUB2
#OEM-Prov-Leaf	3	PKI-2_CRT_OEM_LEAF_VALID, *_ EXPIRED, *_INVALID_CONTENT
#PE_Leaf	0	-

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table 35 — ISO 15118 parameters for Test PKI-6 [private environment (PE)]

PKI name	PKI-6	
PKI parameter name	**PKI parameter value**	**Certificate name**
#V2G-Cas / V2G-CA-ID	0	-
#MO-Cas	0	-
#OEM-Cas	0	-
#PE-Cas / PE-CA-ID	1 / 6	PKI-6_CRT_PE_ROOT_VALID
#CPO-Sub-CA	0	-
#Prov-Sub-CA	0	-
#MO-Sub-CA	0	-
#OEM-Sub-CA	0	-
#CPO-EVSE-Leaf	0	-
#Prov-Leaf	0	-
#MO-Contract-Leaf	0	-
#OEM-Prov-Leaf	0	-
#PE_Leaf	3	PKI-6_CRT_OEM_LEAF_VALID, *_ EXPIRED, *_ INVALID_CONTENT

[V2G4-042] The Test-PKI shall consider all ISO 15118-2 requirements which are summarized in Table 36, to setup the valid certificates and corresponding certificate chains.

Table 36 — ISO 15118-2 requirements for valid PKI setup

ISO 15118-2 PKI Requirement	[V2G2-004], [V2G2-009], [V2G2-010], [V2G2-011], [V2G2-012], [V2G2-878], [V2G2-878], [V2G2-867], [V2G2-869], [V2G2-911], [V2G2-868], [V2G2-932], [V2G2-933]

[V2G4-043] At least one PKI shall have a path length >=4 constraint of the PKI certificate tree.

[V2G4-044] At least one PKI shall have a 10 number of concurrently valid V2G Root Certificates for one Root CA. The certificate shall have the suffix *_VALID_1 to *_VALID_10.

7.7.10 Transport layer security (TLS)

The security concept of ISO 15118-2:2014, 7.3 provides a basic transport-based protection mechanism based on transport layer security (TLS). Hence, the *TLS communication setup handling* using certificates (see 7.7.8) is considered by the ATS. Since the verification of the TLS implementation itself as well as associated cipher suites are out of the scope of this document, the test cases focus only the *TLS communication setup.*

In order to test the setup of a protected TCP/IP based communication using a unilaterally authenticated TLS channel between EVCC and SECC, an EVCC_Tester or SECC_Teser needs a set of certificates, keys and certificate chains. These credentials are handled as PICS/PIXIT and are summarized based on PKI-1 in Table 37 and Table 38. Furthermore, some test case will also modify these PICS/PIXIT in order to verify the SUT behavior using *manipulated credentials*. The PICS/PIXITs are a prerequisite before test case execution can be performed. The process to provide these Information to the SUT is out of the scope of this document.

 © ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table 37 — Security Credentials EVCC_Tester TLS Communication Setup based on PKI-1

Certificates	Keys	Certificate Chain
PKI-1_CRT_EVSE_LEAF_VALID, *_ EXPIRED	PKI-1_CRT_ EVSE_LEAF_*_ Private_Key	PKI-1_CRT_CPO_SUB1_VALID -> PKI-1_CRT_CPO_SUB2_VALID
PKI-1_CRT_EVSE_LEAF_VALID	PKI-1_CRT_ EVSE_LEAF_VALID_ Private_Key	PKI-1_CRT_CPO_SUB2_VALID -> PKI-1_CRT_CPO_SUB1_VALID (Invalid chain order)

Table 38 — Security Credentials SECC_Tester TLS Communication Setup based on PKI-1

Certificates	Keys	Certificate Chain
PKI-1_CRT_ V2G_ROOT_VALID	PKI-1_CRT_ EVSE_LEAF_VALID_ Public_Key	PKI-1_CRT_CPO_SUB1_VALID -> PKI-1_CRT_CPO_SUB2_VALID
PKI-1_CRT_ V2G_ROOT_VALID	PKI-1_CRT_ EVSE_LEAF_VALID_ Public_Key	PKI-1_CRT_CPO_SUB2_VALID -> PKI-1_CRT_CPO_SUB1_VALID (Invalid chain order)

Beside the PICSs/PIXITs additional information e.g. used cipher suite or handshake results are required, to handle the *TLS Communication Setup*. These data are structured in Annex P.4. The described data structures are used to setup and control the TLS communication session.

The setup information will be used by the port map operation to configure the TLS connection.

The control information will be sent to the TLS port using the *sent to* directive with destination is set to *local.* This enables to set port control information or get TLS handshake information.

7.7.11 XML security

7.7.11.1 General information

Specific message content is signed on application layer (XML-messages), if data is supposed to be protected on the way from or to a secondary actor, or if the integrity protection has to last longer than the existence of the TLS channel. To test these integrity protection means on application layer, additional external functions are required.

7.7.11.2 Signing and Verification requirements for XML security

[V2G4-045] The external function for XML security shall provide a mean to calculate the message digest using a given digest algorithm and data stream.

[V2G4-046] The external function for XML security shall provide a mean to encode XML data as EXI representation of this stream.

[V2G4-047] The external function for XML security shall provide a mean to sign a data stream based on a signature algorithm.

[V2G4-048] The external function for XML security shall provide a mean to verify a signed data stream. This function shall use a given signature algorithm, a signature, a leaf certificate and the EXI encoded data that should be signed. It shall return whether verification was successful or otherwise.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

[V2G4-049] The external function for XML security shall provide a mean to verify a certificate chain up to the root CA. This function shall use a given certificate chain and a root CA for verification. It shall return whether verification was successful or otherwise.

[V2G4-050] The external function for XML security shall provide a mean to generate a GenChallenge value.

7.7.11.3 ECDH requirements for XML security

[V2G4-051] The external function for XML security shall provide a mean to calculate a shared secret based on ECDH. This function shall use a given certificate and a Diffie-Hellman public key.

[V2G4-052] The external function for XML security shall provide a mean to decrypt a private key based on ECDH. This function shall use a given shared secret and an encrypted private key.

7.7.11.4 Certificate handling requirements for XML security

[V2G4-053] The external function for XML security shall provide a mean to load a certificate encoded as X509CertificateType (refer to Annex P.7). This function shall use a given unique PKI and certificate name according to 7.7.8.

[V2G4-054] The external function for XML security shall provide a mean to load a private key related to a certificate. This function shall use a given unique certificate name according to 7.7.8.

[V2G4-055] The external function for XML security shall provide a mean to load the EMAID of the contract certificate.

7.7.11.5 SA requirements for XML security

[V2G4-056] The external function for XML security shall provide a mean to verify the authenticity of the OEMProvisioning leaf certificate by the SA. It shall return whether verification was successful or otherwise.

[V2G4-057] The external function for XML security shall provide a mean to update the contract certificate by the SA. It shall return whether verification was successful or otherwise.

[V2G4-058] The external function for XML security shall provide a mean to get the SAProvisioning certificate chain from the SA encoded as CertificateChainType (refer to Annex P.7). This function shall use a given OEMProvisioning leaf or contract certificate.

[V2G4-059] The external function for XML security shall provide a mean to get the contract certificate chain encoded from the SA as CertificateChainType (refer to Annex P.7). This function shall use a given OEMProvisioning leaf or contract certificate.

[V2G4-060] The external function for XML security shall provide a mean to get the contract signature encrypted private key from the SA encoded as ContractSignatureEncryptedPrivateKeyType (refer to Annex P.7). This function shall use a given OEMProvisioning leaf or contract certificate.

[V2G4-061] The external function for XML security shall provide a mean to get the Diffie-Hellman public key from the SA encoded as DiffieHellmanPublickeyType (refer to Annex P.7). This function shall use a given OEMProvisioning leaf or contract certificate.

[V2G4-062] The external function for XML security shall provide a mean to get the EMAID of a contract certificate from the SA encoded as EMAIDType (refer to Annex P.7). This function shall use a given OEMProvisioning leaf or contract certificate.

[V2G4-063] The external function for XML security shall provide a mean to get the signature value for the signed message parts ContractSignatureCertChain, ContractSignatureEncryptedPrivateKey,

 © ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

DHpublickey and eMAID (refer to Annex P.7) from the SA. This function shall use a given OEMProvisioning leaf or contract certificate.

7.7.12 EIM status control functionality

7.7.12.1 General information

To ensure EIM authorization, the MTC is able to detect and control EIM status information by using special functions depending on the MTC's type (SECC_Tester).

The parallel test component (PTC) for EIM status continuously observes the status of the EIM authorization until the authorization process is successful or failed.

7.7.12.2 Requirements of EIM status control for SECC testing

[V2G4-064] The service function for EIM status control shall provide a mean to initiate the EIM authorization.

[V2G4-065] The service function for EIM status control shall provide a mean to process incoming EIM status information from the EIM status PTC.

7.7.12.3 Requirements of EIM status PTC for SECC testing

[V2G4-066] The EIM status PTC for SECC testing shall provide a function to inform the MTC if the EIM authorization is successful. If no change event occurs within a given time [ms], the function shall report an error.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

8 Test case descriptions for 15118-2 V2GTP

8.1 General information

This clause covers all test case descriptions for the *V2GTP requirements* as specified in ISO 15118-2. Sub clause 8.2 details all test case descriptions, where the *SECC* is the SUT. Sub clause 8.3 details all test case descriptions, where the *EVCC* is the SUT.

8.2 SECC test cases

Table **39 - Test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_001'**

TC Id	TC_SECC_CMN_VTB_V2GTPSessionSetup_001
Test objective	Test System executes GoodCase procedure and sends a SessionSetupReq message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H and 'payloadType' equals to '0x8001'H. (V2GTP Header is matched for V2G message content). Test System then checks that the SUT sends a SessionSetupRes message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H and 'payloadType' equals to '0x8001'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-087], [V2G2-088]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSessionSetup_001	

Table 39 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_001'.

Table **40 - Test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_002'**

TC Id	TC_SECC_CMN_VTB_V2GTPSessionSetup_002
Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadLength' equals to '00000000'H in the V2GTP header of the SessionSetupReq message. Test System then checks that the SUT ignores the SessionSetupReq message by detecting a 'V2G_Msg_Timeout'. If the SUT terminates the TCP connection before 'V2G_Msg_Timeout', the test result shall also be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 40 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_002'.

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **41 - Test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_003'**

TC Id	TC_SECC_CMN_VTB_V2GTPSessionSetup_003
Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadType' equals to '7001'H in the V2GTP header of the SessionSetupReq message. Test System then checks that the SUT ignores the SessionSetupReq message by detecting a 'V2G_Msg_Timeout'. If the SUT terminates the TCP connection before 'V2G_Msg_Timeout', the test result shall also be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 41 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_003'.

Table **42 - Test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_004'**

TC Id	TC_SECC_CMN_VTB_V2GTPSessionSetup_004
Test objective	Test System executes GoodCase procedure and sends an invalid 'protocolVersion' equals to 'FF'H in the V2GTP header of the SessionSetupReq message. Test System then checks that the SUT ignores the SessionSetupReq message by detecting a 'V2G_Msg_Timeout'. If the SUT terminates the TCP connection before 'V2G_Msg_Timeout', the test result shall also be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 42 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **43 - Test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_005'**

TC Id	TC_SECC_CMN_VTB_V2GTPSessionSetup_005
Test objective	Test System executes GoodCase procedure and sends an invalid 'invProtocolVersion' equals to 'FF'H in the V2GTP header of the SessionSetupReq message. Test System then checks that the SUT ignores the SessionSetupReq message by detecting a 'V2G_Msg_Timeout'. If the SUT terminates the TCP connection before 'V2G_Msg_Timeout', the test result shall also be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 43 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSessionSetup_005'.

Table **44 - Test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_001'**

TC Id	TC_SECC_CMN_VTB_V2GTPSDP_001
Test objective	Test System executes GoodCase procedure and sends a valid SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H and 'payloadType' equals to '0x9000'H. (V2GTP Header is matched for SDP message content). Test System then checks that the SUT sends a SDP response message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H, payloadType' equals to '0x9001'H and 'payloadLength' equals to '00000014'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.10.1.3.1; 7.10.1.5
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-087], [V2G2-130], [V2G2-152], [V2G2-153]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSDP_001	

Table 44 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **45 - Test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_002'**

TC Id	TC_SECC_CMN_VTB_V2GTPSDP_002
Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadLength' equals to '00000000'H in the V2GTP header of the SDP request message. Test System then checks that the SUT ignores the SDP response message by detecting an SDP message timeout.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSDP_002	

Table 45 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_002'.

Table **46 - Test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_003'**

TC Id	TC_SECC_CMN_VTB_V2GTPSDP_003
Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadType' equals to '7000'H in the V2GTP header of the SDP request message. Test System then checks that the SUT ignores the SDP response message by detecting an SDP message timeout.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSDP_002	

Table 46 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_003'.

Table **47 - Test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_004'**

TC Id	TC_SECC_CMN_VTB_V2GTPSDP_004
Test objective	Test System executes GoodCase procedure and sends an invalid 'protocolVersion' equals to 'FF'H in the V2GTP header of the SDP request message. Test System then checks that the SUT ignores the SDP response message by detecting an SDP message timeout.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSDP_002	

Table 47 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **48 - Test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_005'**

TC Id	TC_SECC_CMN_VTB_V2GTPSDP_005
Test objective	Test System executes GoodCase procedure and sends an invalid 'invProtocolVersion' equals to 'FF'H in the V2GTP header of the SDP request message. Test System then checks that the SUT ignores the SDP response message by detecting an SDP message timeout.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_V2GTPSDP_002	

Table 48 lists the test case description for 'TC_SECC_CMN_VTB_V2GTPSDP_005'.

8.3 EVCC test cases

Table **49 - Test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_001'**

TC Id	TC_EVCC_CMN_VTB_V2GTPSessionSetup_001
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolRes message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H and 'payloadType' equals to '0x8001'H. (V2GTP Header is matched for V2G message content). Test System then checks that the SUT sends a SessionSetupReq message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H and 'payloadType' equals to '0x8001'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-087], [V2G2-088]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_001	

Table 49 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **50 - Test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_002'**

TC Id	TC_EVCC_CMN_VTB_V2GTPSessionSetup_002
Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadLength' equals to '00000000'H in the V2GTP header of the SessionSetupRes message. Test System then checks that the SUT ignores the SessionSetupRes message by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if the 'V2G_Msg_Timeout' timer has expired. If the SUT terminates the TCP connection before 'V2G_Msg_Timeout', the test result shall also be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 50 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_002'.

Table **51 - Test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_003'**

TC Id	TC_EVCC_CMN_VTB_V2GTPSessionSetup_003
Test objective	Test System executes GoodCase procedure and sends an invalid 'payloadType' equals to '7001'H in the V2GTP header of the SessionSetupRes message. Test System then checks that the SUT ignores the SessionSetupRes message by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if the 'V2G_Msg_Timeout' timer has expired. If the SUT terminates the TCP connection before 'V2G_Msg_Timeout', the test result shall also be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 51 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **52 - Test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_004'**

TC Id	TC_EVCC_CMN_VTB_V2GTPSessionSetup_004
Test objective	Test System executes GoodCase procedure and sends an invalid 'protocolVersion' equals to 'FF'H in the V2GTP header of the SessionSetupRes message. Test System then checks that the SUT ignores the SessionSetupRes message by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if the 'V2G_Msg_Timeout' timer has expired. If the SUT terminates the TCP connection before 'V2G_Msg_Timeout', the test result shall also be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 52 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_004'.

Table **53 - Test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_005'**

TC Id	TC_EVCC_CMN_VTB_V2GTPSessionSetup_005
Test objective	Test System executes GoodCase procedure and sends an invalid 'invProtocolVersion' equals to 'FF'H in the V2GTP header of the SessionSetupRes message. Test System then checks that the SUT ignores the SessionSetupRes message by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if the 'V2G_Msg_Timeout' timer has expired. If the SUT terminates the TCP connection before 'V2G_Msg_Timeout', the test result shall also be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002	

Table 53 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSessionSetup_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **54 - Test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_001'**

TC Id	TC_EVCC_CMN_VTB_V2GTPSDP_001
Test objective	Test System executes GoodCase procedure, starts SDP server and waits for an SDP request message (V2GTP Header is matched for SDP content). Test System then checks that the SUT sends a SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H, 'payloadType' equals to '0x9000'H and 'payloadLength' equals to 2.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.10.1.3.1; 7.10.1.4
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-087], [V2G2-127], [V2G2-140], [V2G2-141]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSDP_001	

Table 54 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_001'.

Table **55 - Test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_002'**

TC Id	TC_EVCC_CMN_VTB_V2GTPSDP_002
Test objective	Test System executes GoodCase procedure and waits for a valid SDP request message. Test System then sends an invalid 'payloadLength' equals to '00000000'H in the V2GTP header of the SDP response message. Test System then checks the repetition of SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H, 'payloadType' equals to '0x9000'H and 'payloadLength' equals to 2.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSDP_002	

Table 55 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **56 - Test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_003'**

TC Id	TC_EVCC_CMN_VTB_V2GTPSDP_003
Test objective	Test System executes GoodCase procedure and waits for a valid SDP request message. Test System then sends an invalid 'payloadType' equals to '7001'H in the V2GTP header of the SDP response message. Test System then checks the repetition of SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H, 'payloadType' equals to '0x9000'H and 'payloadLength' equals to 2.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSDP_002	

Table 56 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_003'.

Table **57 - Test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_004'**

TC Id	TC_EVCC_CMN_VTB_V2GTPSDP_004
Test objective	Test System executes GoodCase procedure and waits for a valid SDP request message. Test System then sends an invalid 'protocolVersion' equals to 'FF'H in the V2GTP header of the SDP response message. Test System then checks the repetition of SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H, 'payloadType' equals to '0x9000'H and 'payloadLength' equals to 2.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSDP_002	

Table 57 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **58 - Test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_005'**

TC Id	TC_EVCC_CMN_VTB_V2GTPSDP_005
Test objective	Test System executes GoodCase procedure and waits for a valid SDP request message. Test System then sends an invalid 'invProtocolVersion' equals to 'FF'H in the V2GTP header of the SDP response message. Test System then checks the repetition of SDP request message with the V2GTP header information 'protocolVersion' equals to '0x01'H, 'invProtocolVersion' equals to 'FE'H, 'payloadType' equals to '0x9000'H and 'payloadLength' equals to 2.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.8.3.1; 7.8.3.2
Referenced requirement(s)	[V2G2-082], [V2G2-086], [V2G2-800]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_V2GTPSDP_002	

Table 58 lists the test case description for 'TC_EVCC_CMN_VTB_V2GTPSDP_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

9 Test case descriptions for 15118-2 SDP messages

9.1 General information

This clause covers all test case descriptions for the *SDP requirements* as specified in ISO 15118-2. Sub clause 9.2 details all test case descriptions, where the *SECC* is the SUT. Sub clause 9.3 details all test case descriptions, where the *EVCC* is the SUT.

9.2 SECC test cases

Table **59 - Test case description for 'TC_SECC_CMN_VTB_SDP_001'**

TC Id	TC_SECC_CMN_VTB_SDP_001
Test objective	Test System executes GoodCase procedure and sends an SDP request message with 'Security' equals to '0x10'H and 'TransportProtocol' equals to '0x00'H. Test System then checks that the SUT sends an SDP response message with 'Security' equals to '0x10'H, 'TransportProtocol' equals to '0x00'H and a valid port and IP address.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.1; 7.10.1.5; 7.10.1.7; 7.4; 8.7.4.4; 8.7.4.3
Referenced requirement(s)	[V2G2-123], [V2G2-147], [V2G2-150], [V2G2-154], [V2G2-624], [V2G2-625], [V2G2-027], [V2G2-144], [V2G2-915], [V2G2-852]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_TLS := false
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_001	

Table 59 lists the test case description for 'TC_SECC_CMN_VTB_SDP_001'.

Table **60 - Test case description for 'TC_SECC_CMN_VTB_SDP_002'**

TC Id	TC_SECC_CMN_VTB_SDP_002
Test objective	Test System executes GoodCase procedure and sends an SDP request message with 'Security' equals to '0x00'H and 'TransportProtocol' equals to '0x00'H. Test System then checks that the SUT sends an SDP response message with 'Security' equals to '0x00'H, 'TransportProtocol' equals to '0x00'H and a valid port and IP address.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.1; 7.10.1.5; 7.10.1.7; 7.4; 8.7.4.4; 8.7.4.3
Referenced requirement(s)	[V2G2-123], [V2G2-147], [V2G2-150], [V2G2-154], [V2G2-624], [V2G2-625], [V2G2-027], [V2G2-144], [V2G2-915], [V2G2-626], [V2G2-852]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_001	

Table 60 lists the test case description for 'TC_SECC_CMN_VTB_SDP_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **61 - Test case description for 'TC_SECC_CMN_VTB_SDP_003'**

TC Id	TC_SECC_CMN_VTB_SDP_003
Test objective	Test System executes GoodCase procedure and sends an SDP request message with 'Security' equals to '0x00'H and 'TransportProtocol' equals to '0x00'H. Test System then checks that the SUT sends an SDP response message with 'Security' equals to '0x00'H, 'TransportProtocol' equals to '0x00'H and a valid port and IP address.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.1; 7.10.1.5; 7.10.1.7; 7.4; 7.3.4; 8.7.4.4
Referenced requirement(s)	[V2G2-123], [V2G2-147], [V2G2-150], [V2G2-154], [V2G2-624], [V2G2-626], [V2G2-027], [V2G2-144], [V2G2-631], [V2G2-915]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_001	

Table 61 lists the test case description for 'TC_SECC_CMN_VTB_SDP_003'.

Table **62 - Test case description for 'TC_SECC_CMN_VTB_SDP_004'**

TC Id	TC_SECC_CMN_VTB_SDP_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_SECC_CommunicationSetup_Timeout' timer has expired on SUT side. Test System then checks that the SUT stops the SDP server and does not respond to a valid SDP request message with 'Security' equals to '0x10'H and 'TransportProtocol' equals to '0x00'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.7.3.1
Referenced requirement(s)	[V2G2-723], [V2G2-029], [V2G2-605]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_TLS := false
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_002	

Table 62 lists the test case description for 'TC_SECC_CMN_VTB_SDP_004'.

Table **63 - Test case description for 'TC_SECC_CMN_VTB_SDP_005'**

TC Id	TC_SECC_CMN_VTB_SDP_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_SECC_CommunicationSetup_Timeout' timer has expired on SUT side. Test System then checks that the SUT stops the SDP server and does not respond to a valid SDP request message with 'Security' equals to '0x00'H and 'TransportProtocol' equals to '0x00'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.7.3.1
Referenced requirement(s)	[V2G2-723], [V2G2-029], [V2G2-605]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_002	

Table 63 lists the test case description for 'TC_SECC_CMN_VTB_SDP_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **64 - Test case description for 'TC_SECC_CMN_VTB_SDP_006'**

TC Id	TC_SECC_CMN_VTB_SDP_006
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_SECC_CommunicationSetup_Timeout' timer has expired on SUT side. Test System then checks that the SUT stops the SDP server and does not respond to a valid SDP request message with 'Security' equals to '0x00'H and 'TransportProtocol' equals to '0x00'H.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.7.3.1
Referenced requirement(s)	[V2G2-723], [V2G2-029], [V2G2-605]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_002	

Table 64 lists the test case description for 'TC_SECC_CMN_VTB_SDP_006'.

Table **65 - Test case description for 'TC_SECC_CMN_VTB_SDP_007'**

TC Id	TC_SECC_CMN_VTB_SDP_007
Test objective	Test System executes GoodCase procedure and sends an SDP request message with 'Security' equals to '0x00'H and 'TransportProtocol' equals to '0x00'H. Test System then checks that the SUT sends an SDP response message with 'Security' equals to '0x10'H, 'TransportProtocol' equals to '0x00'H and a valid port and IP address.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.1; 7.10.1.5; 7.10.1.7; 7.4; 8.7.4.4; 8.7.4.3
Referenced requirement(s)	[V2G2-123], [V2G2-147], [V2G2-150], [V2G2-154], [V2G2-624], [V2G2-027], [V2G2-144], [V2G2-915], [V2G2-627], [V2G2-852]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_TLS := false
PreCondition	
f_SECC_CMN_PR_StateB_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SDP_003	

Table 65 lists the test case description for 'TC_SECC_CMN_VTB_SDP_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

9.3 EVCC test cases

9.3.1 Common test cases

Table **66 - Test case description for 'TC_EVCC_CMN_VTB_SDP_001'**

TC Id	TC_EVCC_CMN_VTB_SDP_001
Test objective	Test System executes GoodCase procedure, starts SDP server and waits for an SDP request message. Test System then checks that the SUT sends an SDP request message with 'Security' equals to '0x10' and 'TransportProtocol' equals to '0x00'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.4
Referenced requirement(s)	[V2G2-142], [V2G2-018], [V2G2-019], [V2G2-623]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_001	

Table 66 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_001'.

Table **67 - Test case description for 'TC_EVCC_CMN_VTB_SDP_002'**

TC Id	TC_EVCC_CMN_VTB_SDP_002
Test objective	Test System executes GoodCase procedure, starts SDP server and waits for an SDP request message. Test System then checks that the SUT sends an SDP request message with 'Security' equals to '0x00' and 'TransportProtocol' equals to '0x00'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.4
Referenced requirement(s)	[V2G2-142], [V2G2-018], [V2G2-019], [V2G2-623]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_001	

Table 67 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **68 - Test case description for 'TC_EVCC_CMN_VTB_SDP_003'**

TC Id	TC_EVCC_CMN_VTB_SDP_003
Test objective	Test System executes GoodCase procedure, starts SDP server and waits for an SDP request message. Test System then checks that the SUT sends an SDP request message with a valid security option and 'TransportProtocol' equals to '0x00'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.4
Referenced requirement(s)	[V2G2-142], [V2G2-018], [V2G2-019], [V2G2-623]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_001	

Table 68 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_003'.

Table **69 - Test case description for 'TC_EVCC_CMN_VTB_SDP_004'**

TC Id	TC_EVCC_CMN_VTB_SDP_004
Test objective	Test System executes GoodCase procedure, starts SDP server and waits for an SDP request message. Test System then checks that the SUT sends an SDP request message with 'Security' '0x00' and 'TransportProtocol' equals to '0x00'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.4; 7.3.4
Referenced requirement(s)	[V2G2-142], [V2G2-018], [V2G2-019], [V2G2-623], [V2G2-630]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_001	

Table 69 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_004'.

Table **70 - Test case description for 'TC_EVCC_CMN_VTB_SDP_005'**

TC Id	TC_EVCC_CMN_VTB_SDP_005
Test objective	Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 70 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **71 - Test case description for 'TC_EVCC_CMN_VTB_SDP_006'**

TC Id	TC_EVCC_CMN_VTB_SDP_006
Test objective	Test System executes GoodCase procedure, starts SDP server and waits for at least two SDP request messages and sends asynchronously two SDP response messages. Test System then checks if the SUT is able to handle an asynchronously arriving SECC Discovery Response message by waiting for a valid SupportedAppProtocolReq on the desired TCP link.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.4; 15118:7.10.1.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-018], [V2G2-019], [V2G2-126], [V2G2-159]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_003	

Table 71 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_006'.

Table **72 - Test case description for 'TC_EVCC_CMN_VTB_SDP_007'**

TC Id	TC_EVCC_CMN_VTB_SDP_007
Test objective	Test System executes GoodCase procedure and sends an SDP request message with a valid security option and 'TransportProtocol' equals to '0x00'. Test System then checks that the SUT does not respond to an SDP request message from the SDP server.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.5
Referenced requirement(s)	[V2G2-145]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_StateB_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_004	

Table 72 lists the test case description for 'TC_EVCC_CMN_VTB_SDP_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

9.3.2 AC specific test cases

Table **73 - Test case description for 'TC_EVCC_AC_VTB_SDP_001'**

TC Id	TC_EVCC_AC_VTB_SDP_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 73 lists the test case description for 'TC_EVCC_AC_VTB_SDP_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **74 - Test case description for 'TC_EVCC_AC_VTB_SDP_002'**

TC Id	TC_EVCC_AC_VTB_SDP_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 74 lists the test case description for 'TC_EVCC_AC_VTB_SDP_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **75 - Test case description for 'TC_EVCC_AC_VTB_SDP_003'**

TC Id	TC_EVCC_AC_VTB_SDP_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 75 lists the test case description for 'TC_EVCC_AC_VTB_SDP_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **76 - Test case description for 'TC_EVCC_AC_VTB_SDP_004'**

TC Id	TC_EVCC_AC_VTB_SDP_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 76 lists the test case description for 'TC_EVCC_AC_VTB_SDP_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **77 - Test case description for 'TC_EVCC_AC_VTB_SDP_005'**

TC Id	TC_EVCC_AC_VTB_SDP_005
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 77 lists the test case description for 'TC_EVCC_AC_VTB_SDP_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **78 - Test case description for 'TC_EVCC_AC_VTB_SDP_006'**

TC Id	TC_EVCC_AC_VTB_SDP_006
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 78 lists the test case description for 'TC_EVCC_AC_VTB_SDP_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **79 - Test case description for 'TC_EVCC_AC_VTB_SDP_007'**

TC Id	TC_EVCC_AC_VTB_SDP_007
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 79 lists the test case description for 'TC_EVCC_AC_VTB_SDP_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **80 - Test case description for 'TC_EVCC_AC_VTB_SDP_008'**

TC Id	TC_EVCC_AC_VTB_SDP_008
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 80 lists the test case description for 'TC_EVCC_AC_VTB_SDP_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **81 - Test case description for 'TC_EVCC_AC_VTB_SDP_009'**

TC Id	TC_EVCC_AC_VTB_SDP_009
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 81 lists the test case description for 'TC_EVCC_AC_VTB_SDP_009'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **82 - Test case description for 'TC_EVCC_AC_VTB_SDP_010'**

TC Id	TC_EVCC_AC_VTB_SDP_010
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 82 lists the test case description for 'TC_EVCC_AC_VTB_SDP_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **83 - Test case description for 'TC_EVCC_AC_VTB_SDP_011'**

TC Id	TC_EVCC_AC_VTB_SDP_011
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 83 lists the test case description for 'TC_EVCC_AC_VTB_SDP_011'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **84 - Test case description for 'TC_EVCC_AC_VTB_SDP_012'**

TC Id	TC_EVCC_AC_VTB_SDP_012
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 84 lists the test case description for 'TC_EVCC_AC_VTB_SDP_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **85 - Test case description for 'TC_EVCC_AC_VTB_SDP_013'**

TC Id	TC_EVCC_AC_VTB_SDP_013
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 85 lists the test case description for 'TC_EVCC_AC_VTB_SDP_013'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **86 - Test case description for 'TC_EVCC_AC_VTB_SDP_014'**

TC Id	TC_EVCC_AC_VTB_SDP_014
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 86 lists the test case description for 'TC_EVCC_AC_VTB_SDP_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **87 - Test case description for 'TC_EVCC_AC_VTB_SDP_015'**

TC Id	TC_EVCC_AC_VTB_SDP_015
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 87 lists the test case description for 'TC_EVCC_AC_VTB_SDP_015'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **88 - Test case description for 'TC_EVCC_AC_VTB_SDP_016'**

TC Id	TC_EVCC_AC_VTB_SDP_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 88 lists the test case description for 'TC_EVCC_AC_VTB_SDP_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

9.3.3 DC specific test cases

Table **89 - Test case description for 'TC_EVCC_DC_VTB_SDP_001'**

TC Id	TC_EVCC_DC_VTB_SDP_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 89 lists the test case description for 'TC_EVCC_DC_VTB_SDP_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **90 - Test case description for 'TC_EVCC_DC_VTB_SDP_002'**

TC Id	TC_EVCC_DC_VTB_SDP_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 90 lists the test case description for 'TC_EVCC_DC_VTB_SDP_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **91 - Test case description for 'TC_EVCC_DC_VTB_SDP_003'**

TC Id	TC_EVCC_DC_VTB_SDP_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 91 lists the test case description for 'TC_EVCC_DC_VTB_SDP_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **92 - Test case description for 'TC_EVCC_DC_VTB_SDP_004'**

TC Id	TC_EVCC_DC_VTB_SDP_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 92 lists the test case description for 'TC_EVCC_DC_VTB_SDP_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **93 - Test case description for 'TC_EVCC_DC_VTB_SDP_005'**

TC Id	TC_EVCC_DC_VTB_SDP_005
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 93 lists the test case description for 'TC_EVCC_DC_VTB_SDP_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **94 - Test case description for 'TC_EVCC_DC_VTB_SDP_006'**

TC Id	TC_EVCC_DC_VTB_SDP_006
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 94 lists the test case description for 'TC_EVCC_DC_VTB_SDP_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **95 - Test case description for 'TC_EVCC_DC_VTB_SDP_007'**

TC Id	TC_EVCC_DC_VTB_SDP_007
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 95 lists the test case description for 'TC_EVCC_DC_VTB_SDP_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **96 - Test case description for 'TC_EVCC_DC_VTB_SDP_008'**

TC Id	TC_EVCC_DC_VTB_SDP_008
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 96 lists the test case description for 'TC_EVCC_DC_VTB_SDP_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **97 - Test case description for 'TC_EVCC_DC_VTB_SDP_009'**

TC Id	TC_EVCC_DC_VTB_SDP_009
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_setPwmMode, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 97 lists the test case description for 'TC_EVCC_DC_VTB_SDP_009'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **98 - Test case description for 'TC_EVCC_DC_VTB_SDP_010'**

TC Id	TC_EVCC_DC_VTB_SDP_010
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 98 lists the test case description for 'TC_EVCC_DC_VTB_SDP_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **99 - Test case description for 'TC_EVCC_DC_VTB_SDP_011'**

TC Id	TC_EVCC_DC_VTB_SDP_011
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 99 lists the test case description for 'TC_EVCC_DC_VTB_SDP_011'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **100 - Test case description for 'TC_EVCC_DC_VTB_SDP_012'**

TC Id	TC_EVCC_DC_VTB_SDP_012
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 100 lists the test case description for 'TC_EVCC_DC_VTB_SDP_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **101 - Test case description for 'TC_EVCC_DC_VTB_SDP_013'**

TC Id	TC_EVCC_DC_VTB_SDP_013
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 101 lists the test case description for 'TC_EVCC_DC_VTB_SDP_013'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **102 - Test case description for 'TC_EVCC_DC_VTB_SDP_014'**

TC Id	TC_EVCC_DC_VTB_SDP_014
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 102 lists the test case description for 'TC_EVCC_DC_VTB_SDP_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **103 - Test case description for 'TC_EVCC_DC_VTB_SDP_015'**

TC Id	TC_EVCC_DC_VTB_SDP_015
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 103 lists the test case description for 'TC_EVCC_DC_VTB_SDP_015'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **104 - Test case description for 'TC_EVCC_DC_VTB_SDP_016'**

TC Id	TC_EVCC_DC_VTB_SDP_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure, starts SDP server and counts the number of SDP request messages without sending an SDP response message. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT repeats sending SDP request messages '50' times. The SUT shall stop sending SDP requests after 50 iterations. Furthermore the Test System checks the reset of the message counter after the previous SDP process.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.10.1.4; 7.10.1.6; 7.4; 8.7.4.2
Referenced requirement(s)	[V2G2-142], [V2G2-623], [V2G2-157], [V2G2-159], [V2G2-160], [V2G2-161], [V2G2-018], [V2G2-019], [V2G2-158], [V2G2-731]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_PausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SDP_002	

Table 104 lists the test case description for 'TC_EVCC_DC_VTB_SDP_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10 Test case descriptions for 15118-2 V2G application layer messages

10.1 General information

This clause covers all test case descriptions for the *V2G Application layer message requirements* as specified in ISO 15118-2.

10.2 SECC test cases

10.2.1 General information

This sub clause covers all V2G Application layer test case descriptions, where the SUT is the *SECC*.

10.2.2 V2G protocol handshake

10.2.2.1 General information

This sub clause covers the *V2G protocol handshake* test case descriptions, where the SUT is the SECC.

10.2.2.2 SECC test cases for SupportedAppProtocol

Table **105 - Test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_001'**

TC Id	TC_SECC_CMN_VTB_SupportedAppProtocol_001
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters. Test System then checks that the SUT sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation' or 'OK_SuccessfulNegotiationWithMinorDeviation' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.2.1; 8.2.2; 8.2.3; 8.8.4.3.1; 7.4; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.4.1; 8.6.2.4.2; 7.3.4; 8.6.2.1; 8.7.4.4; 7.7.3.4; 7.10.1.7; 7.7.3.3; 8.7.4.3
Referenced requirement(s)	[V2G2-165], [V2G2-168], [V2G2-169], [V2G2-176], [V2G2-178], [V2G2-541], [V2G2-026], [V2G2-030], [V2G2-031], [V2G2-033], [V2G2-377], [V2G2-381], [V2G2-391], [V2G2-395], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-721], [V2G2-602], [V2G2-626], [V2G2-665], [V2G2-068], [V2G2-871], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	
PIXIT selection	PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_Binding_001	
Expected behavior	
f_SECC_CMN_PR_InitConfiguration_002, f_SECC_CMN_TB_VTB_SupportedAppProtocol_001	

Table 105 lists the test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **106 - Test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_002'**

TC Id	TC_SECC_CMN_VTB_SupportedAppProtocol_002
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolReq message with an invalid ISO namespace and all additional mandatory parameters. Test System then checks that the SUT sends a SupportedAppProtocolRes message with ResponseCode 'FAILED_NoNegotiation' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.2.1; 8.2.2; 8.2.3; 8.8.3.1; 7.4; 7.7.3.3
Referenced requirement(s)	[V2G2-165], [V2G2-168], [V2G2-172], [V2G2-176], [V2G2-178], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-026], [V2G2-030], [V2G2-031], [V2G2-033], [V2G2-721], [V2G2-068]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Binding_001	
Expected behavior	
f_SECC_CMN_PR_InitConfiguration_002, f_SECC_CMN_TB_VTB_SupportedAppProtocol_002	

Table 106 lists the test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_002'.

Table **107 - Test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_004'**

TC Id	TC_SECC_CMN_VTB_SupportedAppProtocol_004
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters. Test System then checks that the SUT sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation' or 'OK_SuccessfulNegotiationWithMinorDeviation' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.2.1; 8.2.2; 8.2.3; 8.8.4.3.1; 7.4; 8.6.2.3.1; 7.3.4; 8.6.2.1; 8.7.4.4; 7.7.3.3; 8.7.4.3
Referenced requirement(s)	[V2G2-165], [V2G2-168], [V2G2-169], [V2G2-176], [V2G2-178], [V2G2-541], [V2G2-026], [V2G2-030], [V2G2-031], [V2G2-033], [V2G2-377], [V2G2-391], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-721], [V2G2-665], [V2G2-068], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_TLS := false
PreCondition	
f_SECC_CMN_PR_Binding_001	
Expected behavior	
f_SECC_CMN_PR_InitConfiguration_002, f_SECC_CMN_TB_VTB_SupportedAppProtocol_001	

Table 107 lists the test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **108 - Test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_005'**

TC Id	TC_SECC_CMN_VTB_SupportedAppProtocol_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_SECC_CommunicationSetup_Timeout' timer has expired on SUT side. Test System then checks that the SUT stops waiting for establishing the TCP/TLS connection and does not respond to a TCP/TLS handshake request.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.7.3.1
Referenced requirement(s)	[V2G2-032], [V2G2-605]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Binding_001	
Expected behavior	
f_SECC_CMN_PR_InitConfiguration_002	

Table 108 lists the test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_005'.

Table **109 - Test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_006'**

TC Id	TC_SECC_CMN_VTB_SupportedAppProtocol_006
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolReq message with a list of valid AppProtocols (VersionNumberMinor does not match VersionNumberMinor of the SUT) including ISO namespace and all additional mandatory parameters. Test System then checks that the SUT sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiationWithMinorDeviation' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.2.1; 8.2.2; 8.2.3; 8.8.4.3.1; 7.4; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.4.1; 8.6.2.4.2; 7.3.4; 8.6.2.1; 8.7.4.4; 7.7.3.4; 7.10.1.7; 7.7.3.3; 8.7.4.3
Referenced requirement(s)	[V2G2-165], [V2G2-168], [V2G2-169], [V2G2-176], [V2G2-178], [V2G2-541], [V2G2-026], [V2G2-030], [V2G2-031], [V2G2-033], [V2G2-377], [V2G2-381], [V2G2-391], [V2G2-395], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-721], [V2G2-602], [V2G2-626], [V2G2-665], [V2G2-068], [V2G2-871], [V2G2-170], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	
PIXIT selection	PIXIT_SECC_CMN_VersionNumberMinor
PreCondition	
f_SECC_CMN_PR_Binding_001	
Expected behavior	
f_SECC_CMN_PR_InitConfiguration_002, f_SECC_CMN_TB_VTB_SupportedAppProtocol_004	

Table 109 lists the test case description for 'TC_SECC_CMN_VTB_SupportedAppProtocol_006'.

10.2.3 V2G messages

10.2.3.1 General information

This sub clause includes all *V2G message* test case descriptions, where the SUT is the SECC.

10.2.3.2 SECC test cases for SessionSetup

10.2.3.3 General information

The following sub clauses include all *V2G SessionSetup message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

10.2.3.3.1 Common test cases

Table **110 - Test case description for 'TC_SECC_CMN_VTB_SessionSetup_001'**

TC Id	TC_SECC_CMN_VTB_SessionSetup_001
Test objective	Test System executes GoodCase procedure and sends a SessionSetupReq message with SessionID '0', valid EvccID and all additional mandatory parameters. Test System then checks that the SUT sends a SessionSetupRes message with a new SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.4.1; 8.6.2.4.2; 7.3.4; 8.6.2.1; 8.7.4.4; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-750], [V2G2-377], [V2G2-381], [V2G2-391], [V2G2-395], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-192], [V2G2-665], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_001	

Table 110 lists the test case description for 'TC_SECC_CMN_VTB_SessionSetup_001'.

Table **111 - Test case description for 'TC_SECC_CMN_VTB_SessionSetup_002'**

TC Id	TC_SECC_CMN_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a SessionSetupReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.2; 8.4.3.2.1; 8.4.3.3.3; 8.8.3.1; 8.8.4.3; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_002	

Table 111 lists the test case description for 'TC_SECC_CMN_VTB_SessionSetup_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **112 - Test case description for 'TC_SECC_CMN_VTB_SessionSetup_004'**

TC Id	TC_SECC_CMN_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_SECC_CommunicationSetup_Timeout' timer has expired after receiving a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.3.1; 8.7.3.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-714], [V2G2-715], [V2G2-716], [V2G2-605]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_004	

Table 112 lists the test case description for 'TC_SECC_CMN_VTB_SessionSetup_004'.

10.2.3.3.2 AC specific test cases

Table **113 - Test case description for 'TC_SECC_AC_VTB_SessionSetup_001'**

TC Id	TC_SECC_AC_VTB_SessionSetup_001
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a SessionSetupReq message with the previous sessionID, valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a SessionSetupRes message with the previous SessionID, ResponseCode 'OK_OldSessionJoined', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.3.1; 8.6.2.3.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-377], [V2G2-381], [V2G2-463], [V2G2-741], [V2G2-745], [V2G2-753], [V2G2-754], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_005	

Table 113 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_001'.

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **114 - Test case description for 'TC_SECC_AC_VTB_SessionSetup_002'**

TC Id	TC_SECC_AC_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a SessionSetupReq message the with a new SessionID, valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.3.1; 8.6.2.3.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-377], [V2G2-381], [V2G2-756], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

Table 114 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **115 - Test case description for 'TC_SECC_AC_VTB_SessionSetup_003'**

TC Id	TC_SECC_AC_VTB_SessionSetup_003
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a SessionSetupReq message with a SessionID '0', valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.3.1; 8.6.2.3.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-377], [V2G2-381], [V2G2-750], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

Table 115 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_003'.

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **116 - Test case description for 'TC_SECC_AC_VTB_SessionSetup_004'**

TC Id	TC_SECC_AC_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SessionSetupReq message with the previous sessionID, valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a SessionSetupRes message with the previous SessionID, ResponseCode 'OK_OldSessionJoined', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.3.1; 8.6.2.3.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-377], [V2G2-381], [V2G2-463], [V2G2-741], [V2G2-745], [V2G2-753], [V2G2-754], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_005	

Table 116 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **117 - Test case description for 'TC_SECC_AC_VTB_SessionSetup_005'**

TC Id	TC_SECC_AC_VTB_SessionSetup_005
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SessionSetupReq message the with a new SessionID, valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.3.1; 8.6.2.3.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-377], [V2G2-381], [V2G2-756], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

Table 117 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_005'.

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **118 - Test case description for 'TC_SECC_AC_VTB_SessionSetup_006'**

TC Id	TC_SECC_AC_VTB_SessionSetup_006
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SessionSetupReq message with a SessionID '0', valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.3.1; 8.6.2.3.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-377], [V2G2-381], [V2G2-750], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

Table 118 lists the test case description for 'TC_SECC_AC_VTB_SessionSetup_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.3.3 DC specific test cases

Table **119 - Test case description for 'TC_SECC_DC_VTB_SessionSetup_001'**

TC Id	`TC_SECC_DC_VTB_SessionSetup_001`
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a SessionSetupReq message with the previous sessionID, valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a SessionSetupRes message with the previous SessionID, ResponseCode 'OK_OldSessionJoined', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.4.1; 8.6.2.4.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-391], [V2G2-395], [V2G2-463], [V2G2-741], [V2G2-745], [V2G2-753], [V2G2-754], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	
PreCondition	
`f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001`	
Expected behavior	
`f_SECC_CMN_TB_VTB_SessionSetup_005`	

Table 119 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **120 - Test case description for 'TC_SECC_DC_VTB_SessionSetup_002'**

TC Id	TC_SECC_DC_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a SessionSetupReq message the with a new SessionID, valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.4.1; 8.6.2.4.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-391], [V2G2-395], [V2G2-756], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

Table 120 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **121 - Test case description for 'TC_SECC_DC_VTB_SessionSetup_003'**

TC Id	`TC_SECC_DC_VTB_SessionSetup_003`
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a SessionSetupReq message with a SessionID '0', valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.4.1; 8.6.2.4.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-391], [V2G2-395], [V2G2-750], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	
PreCondition	
`f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001`	
Expected behavior	
`f_SECC_CMN_TB_VTB_SessionSetup_006`	

Table 121 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **122 - Test case description for 'TC_SECC_DC_VTB_SessionSetup_004'**

TC Id	TC_SECC_DC_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SessionSetupReq message with the previous sessionID, valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a SessionSetupRes message with the previous SessionID, ResponseCode 'OK_OldSessionJoined', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.4.1; 8.6.2.4.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-391], [V2G2-395], [V2G2-463], [V2G2-741], [V2G2-745], [V2G2-753], [V2G2-754], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_005	

Table 122 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **123 - Test case description for 'TC_SECC_DC_VTB_SessionSetup_005'**

TC Id	`TC_SECC_DC_VTB_SessionSetup_005`
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SessionSetupReq message the with a new SessionID, valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.4.1; 8.6.2.4.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-391], [V2G2-395], [V2G2-756], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	
PreCondition	
`f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001`	
Expected behavior	
`f_SECC_CMN_TB_VTB_SessionSetup_006`	

Table 123 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **124 - Test case description for 'TC_SECC_DC_VTB_SessionSetup_006'**

TC Id	TC_SECC_DC_VTB_SessionSetup_006
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SessionSetupReq message with a SessionID '0', valid EvccID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a SessionSetupRes message with the current SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.2.1; 8.8.3.1; 8.8.4.3; 8.6.2.4.1; 8.6.2.4.2; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-190], [V2G2-191], [V2G2-543], [V2G2-462], [V2G2-391], [V2G2-395], [V2G2-750], [V2G2-192], [V2G2-725], [V2G2-733]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionSetup_006	

Table 124 lists the test case description for 'TC_SECC_DC_VTB_SessionSetup_006'.

10.2.3.4 SECC test cases for ServiceDiscovery

10.2.3.5 General information

The following sub clauses include all *V2G ServiceDiscovery message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.5.1 Common test cases

Table **125 - Test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_001'**

TC Id	TC_SECC_CMN_VTB_ServiceDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption, valid SupportedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.2; 8.4.3.3.3; 8.8.4.3.1; 8.5.2.9; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.5.2.4; 8.7.4.4; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-265], [V2G2-266], [V2G2-267], [V2G2-268], [V2G2-271], [V2G2-272], [V2G2-282], [V2G2-283], [V2G2-418], [V2G2-545], [V2G2-752], [V2G2-377], [V2G2-381], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-632], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-757], [V2G2-758], [V2G2-759], [V2G2-915], [V2G2-665], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SessionSetup_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_001	

Table 125 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_001'.

Table **126 - Test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_002'**

TC Id	TC_SECC_CMN_VTB_ServiceDiscovery_002
Test objective	Test System executes GoodCase procedure and sends a valid AuthorizationReq message instead of a ServiceDiscoveryReq message. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'..
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.7.2; 8.8.3.1; 8.8.4.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SessionSetup_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_002	

Table 126 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **127 - Test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_003'**

TC Id	TC_SECC_CMN_VTB_ServiceDiscovery_003
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryReq message with an invalid SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an ServiceDiscoveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.2; 8.4.3.3.3; 8.8.3.1; 8.8.4.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SessionSetup_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_003	

Table 127 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_003'.

Table **128 - Test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_004'**

TC Id	TC_SECC_CMN_VTB_ServiceDiscovery_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a SessionSetupRes message with a new SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_SessionSetup_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_004	

Table 128 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDiscovery_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.5.2 AC specific test cases

Table **129 - Test case description for 'TC_SECC_AC_VTB_ServiceDiscovery_001'**

TC Id	`TC_SECC_AC_VTB_ServiceDiscovery_001`
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', all additional mandatory parameters and the PaymentOption and ChargeService which was selected by the EVCC in the previous session.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.2; 8.4.3.3.3; 8.8.4.3.1; 8.5.2.9; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.5.2.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-265], [V2G2-266], [V2G2-267], [V2G2-268], [V2G2-271], [V2G2-272], [V2G2-282], [V2G2-283], [V2G2-418], [V2G2-545], [V2G2-752], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-632], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-741], [V2G2-745], [V2G2-757], [V2G2-758], [V2G2-759], [V2G2-665], [V2G2-725]
Config Id	CF_04_001
PICS selection	`PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	
PreCondition	
`f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_SECC_CMN_TB_VTB_ServiceDiscovery_005`	

Table 129 lists the test case description for 'TC_SECC_AC_VTB_ServiceDiscovery_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **130 - Test case description for 'TC_SECC_AC_VTB_ServiceDiscovery_002'**

TC Id	TC_SECC_AC_VTB_ServiceDiscovery_002
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', all additional mandatory parameters and the PaymentOption and ChargeService which was selected by the EVCC in the previous session.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.2; 8.4.3.3.3; 8.8.4.3.1; 8.5.2.9; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.5.2.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-265], [V2G2-266], [V2G2-267], [V2G2-268], [V2G2-271], [V2G2-272], [V2G2-282], [V2G2-283], [V2G2-418], [V2G2-545], [V2G2-752], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-632], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-741], [V2G2-745], [V2G2-757], [V2G2-758], [V2G2-759], [V2G2-665], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_005	

Table 130 lists the test case description for 'TC_SECC_AC_VTB_ServiceDiscovery_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.5.3 DC specific test cases

Table **131 - Test case description for 'TC_SECC_DC_VTB_ServiceDiscovery_001'**

TC Id	`TC_SECC_DC_VTB_ServiceDiscovery_001`
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', all additional mandatory parameters and the PaymentOption and ChargeService which was selected by the EVCC in the previous session.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.2; 8.4.3.3.3; 8.8.4.3.1; 8.5.2.9; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.5.2.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-265], [V2G2-266], [V2G2-267], [V2G2-268], [V2G2-271], [V2G2-272], [V2G2-282], [V2G2-283], [V2G2-418], [V2G2-545], [V2G2-752], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-632], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-741], [V2G2-745], [V2G2-757], [V2G2-758], [V2G2-759], [V2G2-665], [V2G2-725]
Config Id	CF_04_001
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	
PreCondition	
`f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_SECC_CMN_TB_VTB_ServiceDiscovery_005`	

Table 131 lists the test case description for 'TC_SECC_DC_VTB_ServiceDiscovery_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **132 - Test case description for 'TC_SECC_DC_VTB_ServiceDiscovery_002'**

TC Id	TC_SECC_DC_VTB_ServiceDiscovery_002
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', all additional mandatory parameters and the PaymentOption and ChargeService which was selected by the EVCC in the previous session.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.2; 8.4.3.3.3; 8.8.4.3.1; 8.5.2.9; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.5.2.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-265], [V2G2-266], [V2G2-267], [V2G2-268], [V2G2-271], [V2G2-272], [V2G2-282], [V2G2-283], [V2G2-418], [V2G2-545], [V2G2-752], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-632], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-741], [V2G2-745], [V2G2-757], [V2G2-758], [V2G2-759], [V2G2-665], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDiscovery_005	

Table 132 lists the test case description for 'TC_SECC_DC_VTB_ServiceDiscovery_002'.

10.2.3.6 SECC test cases for ServiceDetail

10.2.3.6.1 General information

The following sub clauses include all *V2G ServiceDetail message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.6.2 Common test cases

Table **133 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_001'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_001
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID, all additional mandatory parameters and the corresponding service information. If the SUT provides more than one service the Test System sends another valid ServiceDetailReq message and the SUT shall respond to this request message with the corresponding service information again until the required information for each service are received.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.4.2; 8.8.4.3.1; 8.6.2.3.1; 8.6.2.4.1; 8.6.3.5; 8.6.3.6; 8.5.2.21; 8.5.2.22; 8.5.2.23; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-548], [V2G2-379], [V2G2-393], [V2G2-414], [V2G2-416], [V2G2-419], [V2G2-420], [V2G2-421], [V2G2-426], [V2G2-427], [V2G2-343], [V2G2-344], [V2G2-345], [V2G2-346], [V2G2-347], [V2G2-348], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-852]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true, PIXIT_SECC_CMN_InternetAccess := unknown
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_001	

Table 133 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_001'.

Table **134 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_002'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_002
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID, all additional mandatory parameters and the corresponding service information (certificate). If the SUT provides more than one service the Test System sends another valid ServiceDetailReq message and the SUT shall respond to this request message with the corresponding service information again until the required information for each service are received.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.4.2; 8.8.4.3.1; 8.6.2.3.2; 8.6.2.4.2; 8.6.3.5; 8.6.3.6; 8.5.2.21; 8.5.2.22; 8.5.2.23; 8.8.3.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-548], [V2G2-385], [V2G2-397], [V2G2-414], [V2G2-416], [V2G2-419], [V2G2-420], [V2G2-421], [V2G2-426], [V2G2-427], [V2G2-428], [V2G2-343], [V2G2-344], [V2G2-345], [V2G2-346], [V2G2-347], [V2G2-348], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_InternetAccess := unknown
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_001	

Table 134 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **135 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_003'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a ServiceDetailReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.2; 8.4.3.3.3; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-195], [V2G2-196], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_002	

Table 135 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_003'.

Table **136 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_004'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_004
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a ServiceDetailReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.2; 8.4.3.3.3; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-195], [V2G2-196], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_002	

Table 136 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **137 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_005'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_005
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with an invalid SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.2; 8.4.3.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-199], [V2G2-200], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_003	

Table 137 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_005'.

Table **138 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_006'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_006
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with an invalid SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.2; 8.4.3.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-199], [V2G2-200], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_003	

Table 138 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **139 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_007'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_007
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, invalid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_ServiceIDInvalid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.2; 8.4.3.4.2; 8.8.3.1; 8.8.4.3.1; 8.6.3.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-464], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-425]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_004	

Table 139 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_007'.

Table **140 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_008'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_008
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, invalid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_ServiceIDInvalid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.2; 8.4.3.4.2; 8.8.3.1; 8.8.4.3.1; 8.6.3.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-464], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-425]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_004	

Table 140 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **141 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_009'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_009
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_005	

Table 141 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_009'.

Table **142 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_010'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_010
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_005	

Table 142 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **143 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_011'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_011
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID, all additional mandatory parameters and the corresponding service information (internetAccess). If the SUT provides more than one service the Test System sends another valid ServiceDetailReq message and the SUT shall respond to this request message with the corresponding service information again until the required information for each service are received.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.4.2; 8.8.4.3.1; 8.6.2.3.1; 8.6.2.4.1; 8.6.3.5; 8.6.3.6; 8.5.2.21; 8.5.2.22; 8.5.2.23; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-548], [V2G2-379], [V2G2-393], [V2G2-414], [V2G2-416], [V2G2-419], [V2G2-420], [V2G2-421], [V2G2-426], [V2G2-427], [V2G2-343], [V2G2-344], [V2G2-345], [V2G2-346], [V2G2-347], [V2G2-348], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-429], [V2G2-852]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true, PIXIT_SECC_CMN_InternetAccess := internetAccess
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_001	

Table 143 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_011'.

Table **144 - Test case description for 'TC_SECC_CMN_VTB_ServiceDetail_012'**

TC Id	TC_SECC_CMN_VTB_ServiceDetail_012
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID, all additional mandatory parameters and the corresponding service information (internetAccess, certificate). If the SUT provides more than one service the Test System sends another valid ServiceDetailReq message and the SUT shall respond to this request message with the corresponding service information again until the required information for each service are received.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.4.2; 8.8.4.3.1; 8.6.2.3.2; 8.6.2.4.2; 8.6.3.5; 8.6.3.6; 8.5.2.21; 8.5.2.22; 8.5.2.23; 8.8.3.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-752], [V2G2-199], [V2G2-200], [V2G2-548], [V2G2-385], [V2G2-397], [V2G2-414], [V2G2-416], [V2G2-419], [V2G2-420], [V2G2-421], [V2G2-426], [V2G2-427], [V2G2-428], [V2G2-343], [V2G2-344], [V2G2-345], [V2G2-346], [V2G2-347], [V2G2-348], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-429]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_InternetAccess := internetAccess
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ServiceDetail_001	

Table 144 lists the test case description for 'TC_SECC_CMN_VTB_ServiceDetail_012'.

10.2.3.7 SECC test cases for PaymentServiceSelection

10.2.3.7.1 General information

The following sub clauses include all *V2G PaymentServiceSelection message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.7.2 Common test cases

Table **145 - Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_001'**

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_001
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.5.3; 8.8.4.3.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-551], [V2G2-752], [V2G2-377], [V2G2-381], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-636], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	
PIXIT selection	PIXIT_SECC_CMN_VAS := none_
PreCondition	
f_SECC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_001	

Table 145 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_001'.

Table **146 - Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_002'**

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_002
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'ExternalPayment', valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.5.3; 8.8.4.3.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-551], [V2G2-752], [V2G2-377], [V2G2-391], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-852]_eIM
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_SECC_CMN_TLS := true
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_001	

Table 146 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **147 - Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_003'**

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_003
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'Contract', valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.5.3; 8.8.4.3.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-551], [V2G2-752], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_001	

Table 147 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_003'.

Table **148 - Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_004'**

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_004
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PaymentServiceSelectionReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.3.3; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_003
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_002	

Table 148 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **149 - Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_005'**

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_005
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with an invalid SessionID, SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.5.3; 8.8.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_003
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 149 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_005'.

Table **150 - Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_006'**

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_006
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption, with ServiceID value equal to 1, an additional invalid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_ServiceSelectionInvalid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.5.3; 8.8.3.1; 8.6.3.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-752], [V2G2-467], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-433]
Config Id	CF_04_003
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 150 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **151 - Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_007'**

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_007
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption, all additional mandatory parameters and with a ServiceID value equal to 2, but no charge service ID (ServiceID = 1). Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_NoChargeServiceSelected' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.5.3; 8.8.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-752], [V2G2-804], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_003
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_005	

Table 151 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_007'.

Table **152 - Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_008'**

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_008
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_003
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_006	

Table 152 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **153 - Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_009'**

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_009
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'contract', valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_PaymentSelectionInvalid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.5.3; 8.8.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-752], [V2G2-465], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_003
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_SECC_CMN_PaymentOption := externalPayment
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_007	

Table 153 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_009'.

Table **154 - Test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_010'**

TC Id	TC_SECC_CMN_VTB_PaymentServiceSelection_010
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'externalPayment', valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_PaymentSelectionInvalid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.5.3; 8.8.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-203], [V2G2-204], [V2G2-752], [V2G2-465], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_003
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_PaymentOption := contract
PreCondition	
f_SECC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentServiceSelection_007	

Table 154 lists the test case description for 'TC_SECC_CMN_VTB_PaymentServiceSelection_010'.

10.2.3.8 SECC test cases for PaymentDetails

10.2.3.8.1 General information

The following sub clauses include all *V2G PaymentDetails message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

10.2.3.8.2 Common test cases

Table **155 - Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_001'**

TC Id	TC_SECC_CMN_VTB_PaymentDetails_001
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.4.3.6.3; 8.6.2.3.2; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-560], [V2G2-752], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-899], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_001	

Table 155 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_001'.

Table **156 - Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_002'**

TC Id	TC_SECC_CMN_VTB_PaymentDetails_002
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.4.3.6.3; 8.6.2.3.2; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-560], [V2G2-752], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-899], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_001	

Table 156 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **157 - Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_003'**

TC Id	TC_SECC_CMN_VTB_PaymentDetails_003
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.4.3.6.3; 8.6.2.3.2; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-560], [V2G2-752], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-899], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_001	

Table 157 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_003'.

Table **158 - Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_004'**

TC Id	TC_SECC_CMN_VTB_PaymentDetails_004
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PaymentDetailsReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.3.3; 8.4.3.6.3; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_002	

Table 158 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **159 - Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_005'**

TC Id	TC_SECC_CMN_VTB_PaymentDetails_005
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with an invalid SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.4.3.6.3; 8.8.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_003	

Table 159 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_005'.

Table **160 - Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_006'**

TC Id	TC_SECC_CMN_VTB_PaymentDetails_006
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_004	

Table 160 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **161 - Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_007'**

TC Id	TC_SECC_CMN_VTB_PaymentDetails_007
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, invalid contract certificate chain (Expired) and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED_CertificateExpired' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.4.3.6.3; 8.8.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-474], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-899]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_005	

Table 161 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_007'.

Table **162 - Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_008'**

TC Id	TC_SECC_CMN_VTB_PaymentDetails_008
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, invalid contract certificate chain (Invalid content -> Country name was set) and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED', 'fAILED_NoCertificateAvailable' or 'fAILED_CertificateNotAllowedAtThisEVSE' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.4.3.6.3; 8.8.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-899]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_005	

Table 162 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **163 - Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_009'**

TC Id	TC_SECC_CMN_VTB_PaymentDetails_009
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, invalid contract certificate chain (Expired issuer Sub CA 1) and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED', 'fAILED_NoCertificateAvailable' or 'fAILED_CertificateNotAllowedAtThisEVSE' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.4.3.6.3; 8.8.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-899]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_005	

Table 163 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_009'.

Table **164 - Test case description for 'TC_SECC_CMN_VTB_PaymentDetails_010'**

TC Id	TC_SECC_CMN_VTB_PaymentDetails_010
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailsReq message with the current SessionID, valid EMAID, invalid contract certificate chain (Expired issuer Sub CA 2) and all additional mandatory parameters. Test System then checks that the SUT sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED', 'fAILED_NoCertificateAvailable' or 'fAILED_CertificateNotAllowedAtThisEVSE' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.4.3.6.3; 8.8.3.1;
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-208], [V2G2-209], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-899]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PaymentDetails_005	

Table 164 lists the test case description for 'TC_SECC_CMN_VTB_PaymentDetails_010'.

10.2.3.9 SECC test cases for Authorization

10.2.3.9.1 General information

The following sub clauses include all *V2G Authorization message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.9.2 Common test cases

Table **165 - Test case description for 'TC_SECC_CMN_VTB_Authorization_001'**

TC Id	TC_SECC_CMN_VTB_Authorization_001
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a valid AuthorizationRes message with EVSEProcessing 'Ongoing_WaitingForCustomerInteraction' was received before, the Test System sends another AuthorizationReq message. This message sequence will be repeated until the authorization has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.4.3.1; 8.6.2.3.1; 8.6.2.4.1; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-377], [V2G2-391], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_001	

Table 165 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_001'.

Table **166 - Test case description for 'TC_SECC_CMN_VTB_Authorization_002'**

TC Id	TC_SECC_CMN_VTB_Authorization_002
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with a valid signature, current SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a valid AuthorizationRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another AuthorizationReq message. This message sequence will be repeated until the authorization has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.4.3.1; 8.6.2.3.2; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4; 7.3.2; 7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-901], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_002	

Table 166 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **167 - Test case description for 'TC_SECC_CMN_VTB_Authorization_003'**

TC Id	TC_SECC_CMN_VTB_Authorization_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of an AuthorizationReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.3.3; 8.7.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_003	

Table 167 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_003'.

Table **168 - Test case description for 'TC_SECC_CMN_VTB_Authorization_004'**

TC Id	TC_SECC_CMN_VTB_Authorization_004
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of an AuthorizationReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.3.3; 8.7.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-736], [V2G2-539], [V2G2-729], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_003	

Table 168 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **169 - Test case description for 'TC_SECC_CMN_VTB_Authorization_005'**

TC Id	TC_SECC_CMN_VTB_Authorization_005
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the an invalid SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationyRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.3.1 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_004	

Table 169 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_005'.

Table **170 - Test case description for 'TC_SECC_CMN_VTB_Authorization_006'**

TC Id	TC_SECC_CMN_VTB_Authorization_006
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with a valid signature, invalid SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.3.1 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_005	

Table 170 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **171 - Test case description for 'TC_SECC_CMN_VTB_Authorization_007'**

TC Id	TC_SECC_CMN_VTB_Authorization_007
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with a valid signature, current SessionID, invalid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_ChallengeInvalid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.3.1 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-752], [V2G2-475], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-901]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_006	

Table 171 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_007'.

Table **172 - Test case description for 'TC_SECC_CMN_VTB_Authorization_008'**

TC Id	TC_SECC_CMN_VTB_Authorization_008
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_007	

Table 172 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **173 - Test case description for 'TC_SECC_CMN_VTB_Authorization_009'**

TC Id	TC_SECC_CMN_VTB_Authorization_009
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_007	

Table 173 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_009'.

Table **174 - Test case description for 'TC_SECC_CMN_VTB_Authorization_010'**

TC Id	TC_SECC_CMN_VTB_Authorization_010
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with an invalid signature (wrong private key used), current SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.3.1 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-901]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_008	

Table 174 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **175 - Test case description for 'TC_SECC_CMN_VTB_Authorization_011'**

TC Id	TC_SECC_CMN_VTB_Authorization_011
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with an invalid signature (signed body changed after signature calculation), current SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' or 'FAILED_ChallengeInvalid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.3.1 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-901]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_009	

Table 175 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_011'.

Table **176 - Test case description for 'TC_SECC_CMN_VTB_Authorization_012'**

TC Id	TC_SECC_CMN_VTB_Authorization_012
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with a valid signature, current SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a valid AuthorizationRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another AuthorizationReq message. This message sequence will be repeated until the authorization has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.4.3.1; 8.6.2.3.2; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4; 7.3.2; 7.9.2.4.5; 7.9.2.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-820], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_002	

Table 176 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **177 - Test case description for 'TC_SECC_CMN_VTB_Authorization_013'**

TC Id	TC_SECC_CMN_VTB_Authorization_013
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with a valid signature, current SessionID, valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a valid AuthorizationRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another AuthorizationReq message. This message sequence will be repeated until the authorization has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.4.3.1; 8.6.2.3.2; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4; 7.3.2; 7.9.2.4.5; 7.9.2.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-821], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_002	

Table 177 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_013'.

Table **178 - Test case description for 'TC_SECC_CMN_VTB_Authorization_014'**

TC Id	TC_SECC_CMN_VTB_Authorization_014
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing_WaitingForCustomerInteraction' and all additional mandatory parameters. Furthermore the repetition of the Authorization message sequence will be checked (EVSEProcessing := 'Ongoing_WaitingForCustomerInteraction') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT sends a AuthorizationRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'. For this test case the user shall not execute the authorization by a customer interaction to provoke the timeout.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.4.3.1; 8.6.2.3.1; 8.6.2.4.1; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4; 8.7.3.5; 7.4; 8.7.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-377], [V2G2-391], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665], [V2G2-687], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435], [V2G2-854]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM, PICS_SECC_CMN_EIMDone := v2gAuthorization
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_010	

Table 178 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **179 - Test case description for 'TC_SECC_CMN_VTB_Authorization_015'**

TC Id	TC_SECC_CMN_VTB_Authorization_015
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing' and all additional mandatory parameters. Furthermore the repetition of the Authorization message sequence will be checked (EVSEProcessing := 'Ongoing') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT sends a AuthorizationRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.4.3.1; 8.6.2.3.2; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4; 7.3.2; 7.9.2.4.5; 8.7.3.5; 7.4; 8.7.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-901], [V2G2-665], [V2G2-687], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435], [V2G2-855]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_AuthorizationOngoing := ongoing
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_011	

Table 179 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_015'.

Table **180 - Test case description for 'TC_SECC_CMN_VTB_Authorization_016'**

TC Id	TC_SECC_CMN_VTB_Authorization_016
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing' and all additional mandatory parameters. Furthermore the repetition of the Authorization message sequence will be checked (EVSEProcessing := 'Ongoing') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT sends a AuthorizationRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.4.3.1; 8.6.2.3.2; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4; 7.3.2; 7.9.2.4.5; 7.9.2.4.3; 8.7.3.5; 7.4; 8.7.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-820], [V2G2-665], [V2G2-687], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435], [V2G2-855]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_AuthorizationOngoing := ongoing
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_011	

Table 180 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **181 - Test case description for 'TC_SECC_CMN_VTB_Authorization_017'**

TC Id	TC_SECC_CMN_VTB_Authorization_017
Test objective	Test System executes GoodCase procedure and sends an AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing' and all additional mandatory parameters. Furthermore the repetition of the Authorization message sequence will be checked (EVSEProcessing := 'Ongoing') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT sends a AuthorizationRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.2; 8.7.4.2; 8.8.4.3.1; 8.6.2.3.2; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4; 7.3.2; 7.9.2.4.5; 7.9.2.4.3; 8.7.3.5; 7.4; 8.7.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-212], [V2G2-213], [V2G2-563], [V2G2-752], [V2G2-856], [V2G2-381], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-821], [V2G2-665], [V2G2-687], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435], [V2G2-855]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_AuthorizationOngoing := ongoing
PreCondition	
f_SECC_CMN_PR_PaymentDetails_001	
Expected behavior	
f_SECC_CMN_TB_VTB_Authorization_011	

Table 181 lists the test case description for 'TC_SECC_CMN_VTB_Authorization_017'.

10.2.3.10 SECC test cases for ChargeParameterDiscovery

10.2.3.10.1 General information

The following sub clauses include all *V2G ChargeParameterDiscovery message* test case descriptions, where the SUT is the SECC.

10.2.3.10.2 Common test cases

Table **182 - Test case description for 'TC_SECC_CMN_VTB_ChargeParameterDiscovery_001'**

TC Id	TC_SECC_CMN_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a ChargeParameterDiscoveryReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.3.3; 8.4.3.8.3; 8.8.3.1; 8.8.4.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_001	

Table 182 lists the test case description for 'TC_SECC_CMN_VTB_ChargeParameterDiscovery_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **183 - Test case description for 'TC_SECC_CMN_VTB_ChargeParameterDiscovery_002'**

TC Id	TC_SECC_CMN_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_002	
Expected behavior	
f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_002	

Table 183 lists the test case description for 'TC_SECC_CMN_VTB_ChargeParameterDiscovery_002'.

10.2.3.10.3 AC specific test cases

Table **184 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_001'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 184 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **185 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_002'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 185 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_002'.

Table **186 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_003'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_003
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with an invalid SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_002	

Table 186 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **187 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_004'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_004
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, an invalid RequestedEnergyTransferType 'dC_extended' and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongEnergyTransferMode' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-752], [V2G2-476], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_003	

Table 187 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_004'.

Table **188 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_005'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_005
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, valid RequestedEnergyTransferType, EVMaxCurrent '-100' and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongChargeParameter' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-752], [V2G2-477], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_004	

Table 188 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **189 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_006'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_006
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown,
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 189 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **190 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_007'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_007
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 190 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **191 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_008'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_008
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.3; 8.5.2.16; 8.5.2.17; 8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 191 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **192 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_009'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_009
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.5.2.16; 8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 192 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_009'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **193 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_010'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_010
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.3; 8.5.2.16; 8.5.2.17; 8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 193 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **194 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_011'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_011
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 8.5.2.16; 8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 194 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_011'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **195 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_012'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_012
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 195 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **196 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_013'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_013
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.3; 8.5.2.16; 8.5.2.17; 8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 196 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_013'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **197 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_014'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_014
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing', a valid scheduleList and all additional mandatory parameters. Furthermore the repetition of the ChargeParameterDiscovery message sequence will be checked (EVSEProcessing := 'Ongoing') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.3; 8.7.3.5; 7.4; 8.7.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665], [V2G2-688], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing := ongoing
PreCondition	
f_SECC_CMN_PR_Authorization_002	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_008	

Table 197 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **198 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_015'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_015
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.3; 8.5.2.16; 8.5.2.17; 8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 198 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_015'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **199 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_016'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_016
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-931], [V2G2-708], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 199 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **200 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_017'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_017
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 200 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_017'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **201 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_018'**

TC Id	`TC_SECC_AC_VTB_ChargeParameterDiscovery_018`
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	`PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false`
PIXIT selection	`PIXIT_SECC_CMN_SalesTariff := salesTariff`
PreCondition	
`f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001`	
Expected behavior	
`f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007`	

Table 201 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_018'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **202 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_019'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_019
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 202 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_019'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **203 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_020'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_020
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 203 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_020'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **204 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_021'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_021
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 204 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_021'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **205 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_022'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_022
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 205 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_022'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **206 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_023'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_023
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 206 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_023'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **207 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_024'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_024
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.5.2.16; 8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 207 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_024'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **208 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_025'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_025
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 208 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_025'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **209 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_026'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_026
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.5.2.16; 8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 209 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_026'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **210 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_027'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_027
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.5.2.16; 8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 210 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_027'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **211 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_028'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_028
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-832], [V2G2-665], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 211 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_028'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **212 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_029'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_029
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 212 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_029'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **213 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_030'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_030
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 8.5.2.16; 8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 213 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_030'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **214 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_031'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_031
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 214 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_031'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **215 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_032'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_032
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 215 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_032'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **216 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_033'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_033
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 216 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_033'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **217 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_034'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_034
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 217 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_034'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **218 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_035'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_035
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007	

Table 218 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_035'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **219 - Test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_036'**

TC Id	TC_SECC_AC_VTB_ChargeParameterDiscovery_036
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.2; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.3.1; 8.5.3.3; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-573], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-358], [V2G2-359], [V2G2-362], [V2G2-363], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_AC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 219 lists the test case description for 'TC_SECC_AC_VTB_ChargeParameterDiscovery_036'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.10.4 DC specific test cases

Table **220 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_001'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 220 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_001'.

Table **221 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_002'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with an invalid SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_002	

Table 221 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **222 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_003'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_003
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, an invalid RequestedEnergyTransferType 'aC_three_phase_core' and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongEnergyTransferMode' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-752], [V2G2-476], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_003	

Table 222 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_003'.

Table **223 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_004'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_004
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, valid RequestedEnergyTransferType, EVMaximumCurrentLimit '-100' and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongChargeParameter' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-752], [V2G2-477], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_004	

Table 223 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **224 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_005'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_005
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 224 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_005'.

Table **225 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_006'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_006
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 225 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **226 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_007'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_007
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 226 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **227 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_008'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_008
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.5.2.16; 8.5.2.17; 8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 227 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **228 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_009'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_009
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4; 8.5.2.16; 8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 228 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_009'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **229 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_010'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_010
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.5.2.16; 8.5.2.17; 8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 229 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **230 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_011'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_011
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 230 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_011'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **231 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_012'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_012
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 231 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **232 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_013'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_013
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.5.2.16; 8.5.2.17; 8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 232 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_013'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **233 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_014'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_014
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing', a valid scheduleList and all additional mandatory parameters. Furthermore the repetition of the ChargeParameterDiscovery message sequence will be checked (EVSEProcessing := 'Ongoing') until the 'V2G_SECC_Ongoing_Timer' has expired on SUT side. Test system then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.3.5; 7.4; 8.7.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-688], [V2G2-712], [V2G2-713], [V2G2-729], [V2G2-034], [V2G2-435]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing := ongoing
PreCondition	
f_SECC_CMN_PR_Authorization_002	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_008	

Table 233 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **234 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_015'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_015
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.5.2.16; 8.5.2.17; 8.6.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-762]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 234 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_015'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **235 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_016'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_016
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-303], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 235 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **236 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_017'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_017
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 236 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_017'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **237 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_018'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_018
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 237 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_018'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **238 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_019'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_019
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 238 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_019'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **239 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_020'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_020
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 239 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_020'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **240 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_021'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_021
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 240 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_021'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **241 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_022'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_022
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating a BCB toggle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional) and B1/B2 transition. Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 241 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_022'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **242 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_023'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_023
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 242 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_023'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **243 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_024'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_024
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4; 8.5.2.16; 8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 243 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_024'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **244 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_025'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_025
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 244 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_025'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **245 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_026'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_026
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4; 8.5.2.16; 8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 245 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_026'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **246 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_027'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_027
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4; 8.5.2.16; 8.5.2.17
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 246 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_027'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **247 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_028'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_028
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a new ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered) and all additional mandatory parameters. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-303]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 247 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_028'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **248 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_029'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_029
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 248 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_029'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **249 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_030'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_030
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple and DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 249 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_030'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **250 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_031'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_031
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList, all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 250 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_031'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **251 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_032'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_032
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, without MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 251 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_032'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **252 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_033'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_033
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 252 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_033'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **253 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_034'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_034
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, without DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-304], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := false, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 253 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_034'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **254 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_035'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_035
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and salesTariff (MaxEntriesSAScheduleTuple is considered) all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 8.5.2.16; 8.5.2.17; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-307], [V2G2-308], [V2G2-309], [V2G2-905], [V2G2-906], [V2G2-316], [V2G2-612], [V2G2-321], [V2G2-613], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := salesTariff
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007	

Table 254 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_035'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **255 - Test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_036'**

TC Id	TC_SECC_DC_VTB_ChargeParameterDiscovery_036
Test objective	Test System executes GoodCase procedure and initiates a paused V2G communication session by sending a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After receipt of a valid SessionStopRes message, Test System checks that the SUT turns off the oscillator for the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a ChargeParameterDiscoveryReq message the current SessionID, a valid RequestedEnergyTransferType, a MaxEntriesSAScheduleTuple, a DepartureTime and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList (MaxEntriesSAScheduleTuple is considered), all additional mandatory parameters and the SAScheduleTuple which was selected by the EVCC in the previous session. If a valid ChargeParameterDiscoveryRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another ChargeParameterDiscoveryReq message. This message sequence will be repeated until the processing of the charging information has finished or 'V2G_EVCC_Ongoing_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.3; 8.8.4.3.3; 8.5.2.7; 8.5.2.12; 8.5.2.13; 8.5.2.14; 8.5.2.15; 8.5.2.18; 8.5.4.1; 8.5.4.4; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.4.3.8.3; 8.7.4.4; 8.7.4.2; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-218], [V2G2-220], [V2G2-582], [V2G2-752], [V2G2-278], [V2G2-279], [V2G2-294], [V2G2-608], [V2G2-299], [V2G2-609], [V2G2-300], [V2G2-304], [V2G2-310], [V2G2-313], [V2G2-610], [V2G2-611], [V2G2-327], [V2G2-614], [V2G2-331], [V2G2-364], [V2G2-365], [V2G2-372], [V2G2-373], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-672], [V2G2-677], [V2G2-773], [V2G2-915], [V2G2-741], [V2G2-745], [V2G2-832], [V2G2-665], [V2G2-303], [V2G2-725]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_DepartureTime := true, PICS_SECC_CMN_Pause := true, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true, PICS_SECC_CMN_MaxEntriesSAScheduleTuple := true
PIXIT selection	PIXIT_SECC_CMN_SalesTariff := unknown
PreCondition	
f_checkWakeUpConfig, f_SECC_DC_PR_SessionStop_001, f_SECC_startSleepingPhase, f_SECC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 255 lists the test case description for 'TC_SECC_DC_VTB_ChargeParameterDiscovery_036'.

10.2.3.11 SECC test cases for PowerDelivery

10.2.3.11.1 General information

The following sub clauses include all *V2G PowerDelivery message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.11.2 AC specific test cases

Table **256 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_001'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_001
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', valid charging profile and TupleID and all additional mandatory parameters and signals CP State C. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.9.3; 8.7.4.3; 8.8.4.3.2; 8.5.3.1; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-576], [V2G2-752], [V2G2-861], [V2G2-224], [V2G2-358], [V2G2-359], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_EvStateInChargeProgress := e_StateC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_001	

Table 256 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_001'.

Table **257 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_002'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_002
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_' and all additional mandatory parameters and signals CP State D. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.9.3; 8.7.4.3; 8.8.4.3.2; 8.5.3.1; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-576], [V2G2-752], [V2G2-861], [V2G2-224], [V2G2-358], [V2G2-359], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_SECC_CMN_EvStateInChargeProgress := e_StateD
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_001	

Table 257 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **258 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_003'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_003
Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2; 8.4.3.9.3; 8.5.3.1; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-813], [V2G2-861], [V2G2-358], [V2G2-359], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown
PreCondition	
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_001	

Table 258 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_003'.

Table **259 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_004'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_004
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.9.2; 8.4.3.9.3; 8.7.4.3; 8.8.4.3.2; 8.5.3.1; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-576], [V2G2-752], [V2G2-861], [V2G2-358], [V2G2-359], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_001	

Table 259 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **260 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_005'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_005
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters and signals CP State B. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.9.2; 8.4.3.9.3; 8.7.4.3; 8.8.4.3.1; 8.5.3.1; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-568], [V2G2-752], [V2G2-864], [V2G2-358], [V2G2-359], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_001	

Table 260 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_005'.

Table **261 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_006'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_006
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PowerDeliveryReq message with chargeProgressType 'Start_'. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.3; 8.7.4.3; 8.8.3.1; 8.8.4.3.2; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_002	

Table 261 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **262 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_007'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_007
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PowerDeliveryReq message with chargeProgressType 'Stop_'. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.3; 8.7.4.3; 8.8.3.1; 8.8.4.3.2; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_002	

Table 262 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_007'.

Table **263 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_008'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_008
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with an invalid SessionID, ChargeProgress 'Start_', valid charging profile and TupleID and all additional mandatory parameters and signals CP State C or D. Test System then checks that the SUT sends a PowerDeliveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.9.3; 8.7.4.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_002	

Table 263 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **264 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_009'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_009
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with an invalid SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.9.3; 8.7.4.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_003	

Table 264 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_009'.

Table **265 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_010'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_010
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', invalid charging profile (value > first pMax entry received in SAScheduleList), valid TupleID and all additional mandatory parameters and signals CP State C or D. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ChargingProfileInvalid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.9.3; 8.7.4.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-225], [V2G2-478], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_004	

Table 265 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **266 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_011'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_011
Test objective	Test System executes GoodCase procedure, sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', valid charging profile, invalid TupleID and all additional mandatory parameters and signals CP State C or D. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_TariffSelectionInvalid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.9.3; 8.7.4.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-479], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_005	

Table 266 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_011'.

Table **267 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_012'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_012
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PowerDelivery_001	

Table 267 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **268 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_013'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_013
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a valid ChargingStatusRes or MeteringReceipt message. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PowerDelivery_001	

Table 268 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_013'.

Table **269 - Test case description for 'TC_SECC_AC_VTB_PowerDelivery_014'**

TC Id	TC_SECC_AC_VTB_PowerDelivery_014
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renogotiate' and all additional mandatory parameters but no PowerDeliveryReq with ChargeProgress 'Start' was sent before. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.9.3; 8.7.4.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-458], [V2G2-812]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_AC_TB_VTB_PowerDelivery_006	

Table 269 lists the test case description for 'TC_SECC_AC_VTB_PowerDelivery_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

10.2.3.11.3 DC specific test cases

Table **270 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_001'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', valid charging profile and TupleID and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-224], [V2G2-590], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 270 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_001'.

Table **271 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_002'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_002
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. Furthermore Test System signals CP State B after received valid PowerDeliveryRes message. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 271 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **272 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_003'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PowerDeliveryReq message with ChargeProgressType 'Start_'. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.4.3.9.2, 8.4.3.3.3; 8.8.3.1; 8.8.4.2.3; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_002	

Table 272 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_003'.

Table **273 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_004'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_004
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PowerDeliveryReq message with ChargeProgressType 'Stop_'. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.4.3.9.2, 8.4.3.3.3; 8.8.3.1; 8.8.4.2.3; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_002	

Table 273 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **274 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_005'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_005
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with an invalid SessionID, ChargeProgress 'Start_', valid charging profile and TupleID and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.3.1; 8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_003	

Table 274 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_005'.

Table **275 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_006'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_006
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with an invalid SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.3.1; 8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_003	

Table 275 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **276 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_007'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_007
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', invalid charging profile (value > first pMax entry received in SAScheduleList), valid TupleID and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ChargingProfileInvalid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.3.1; 8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-225], [V2G2-478], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_004	

Table 276 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_007'.

Table **277 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_008'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_008
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', valid charging profile, invalid TupleID and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_TariffSelectionInvalid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.3.1; 8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-479], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_005	

Table 277 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **278 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_009'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_009
Test objective	Test System executes GoodCase procedure, runs the pre-charge process and starts the V2G_EVCC_PreCharge_Timer by sending the first PreChargeReq message and waits until the pre-charge process is successful and the EVSEPresentVoltage matches the EV battery voltage including tolerances. After receipt of the last PreChargeRes message with the current SessionID, ResponseCode 'OK'and all additional mandatory parameters, Test System waits at minimum until the 'V2G_EVCC_PreCharge_Timer' and maximum the 'V2G_Sequence_Timeout' timer has expired. Test System then checks inbetween or at latest when the 'V2G_Sequence_Timeout' expires that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PowerDelivery_001	

Table 278 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_009'.

Table **279 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_010'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_010
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a valid CurrentDemandRes or MeteringReceipt message. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_PowerDelivery_001	

Table 279 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **280 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_011'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_011
Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2; 8.4.3.9.3; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-813], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 280 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_011'.

Table **281 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_012'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_012
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_' and all additional mandatory parameters. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.9.2; 8.4.3.9.3; 8.7.4.3; 8.8.4.3.2; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-590], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_RenegotiationLoopIndication < PICS_CMN_CMN_LoopCounter
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 281 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **282 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_013'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_013
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renogotiate' and all additional mandatory parameters but no PowerDeliveryReq with ChargeProgress 'Start' was sent before. Test System then checks that the SUT sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.9.3; 8.7.4.3; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-752], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-458], [V2G2-812]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PreCharge_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_006	

Table 282 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_013'.

Table **283 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_014'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_014
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorCode 'FAILED_RESSTemperatureInhibit', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorCode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.3.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 283 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **284 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_015'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_015
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorCode 'FAILED_EVShiftPosition', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorCode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.3.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 284 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_015'.

Table **285 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_016'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_016
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorCode 'FAILED_ChargerConnectorLockFault', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorCode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.3.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 285 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **286 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_017'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_017
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorCode 'FAILED_EVRESSMalfunction', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorCode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.3.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 286 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_017'.

Table **287 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_018'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_018
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorCode 'FAILED_ChargingCurrentdifferential', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorCode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.3.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 287 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_018'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **288 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_019'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_019
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorCode 'FAILED_ChargingVoltageOutOfRange', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorCode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.3.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 288 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_019'.

Table **289 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_020'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_020
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorCode 'FAILED_ChargingSystemIncompatibility', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorCode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.3.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 289 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_020'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **290 - Test case description for 'TC_SECC_DC_VTB_PowerDelivery_021'**

TC Id	TC_SECC_DC_VTB_PowerDelivery_021
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryReq message with the current SessionID, EVErrorCode 'NoData', ChargeProgressType 'Stop_' and all additional mandatory parameters. Test System then checks that the EVErrorCode not influences the EVSE charging process by receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.3.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-223], [V2G2-226], [V2G2-601], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-881], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_PowerDelivery_001	

Table 290 lists the test case description for 'TC_SECC_DC_VTB_PowerDelivery_021'.

10.2.3.12 SECC test cases for CertificateInstallation

10.2.3.12.1 General information

The following sub clauses include all *V2G CertificateInstallation message* test case descriptions, where the SUT is the SECC.

10.2.3.12.2 Common test cases

Table **291 - Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_001'**

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_001
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with a valid signature, current SessionID, OEMProvisioning certificate and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.11.3; 8.7.3.1; 8.8.4.3; 8.5.2.28; 8.5.2.29; 8.5.2.30; 8.6.2.3.2; 8.6.2.4.2; 8.6.3.3; 8.8.3.1; 8.6.2.1; 8.7.4.4; 7.9.2.4.5; 8.5.2.26; 7.9.2.4.3; 7.3.2; 7.3.3; AnnexF; 7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-274], [V2G2-275], [V2G2-554], [V2G2-752], [V2G2-778], [V2G2-779], [V2G2-780], [V2G2-781], [V2G2-782], [V2G2-783], [V2G2-387], [V2G2-399], [V2G2-410], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-653], [V2G2-353], [V2G2-354], [V2G2-814], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-884], [V2G2-822], [V2G2-665], [V2G2-103], [V2G2-104]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_001	

Table 291 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **292 - Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_002'**

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a CertificateInstallationReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.3.3; 8.7.3.1; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_002	

Table 292 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_002'.

Table **293 - Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_003'**

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_003
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with a valid signature, invalid SessionID, OEMProvisioning certificate and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS @url ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_003	

Table 293 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **294 - Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_004'**

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_004	

Table 294 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_004'.

Table **295 - Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_005'**

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_005
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with an invalid signature (wrong private key used), current SessionID, OEMProvisioning certificate and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_005	

Table 295 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **296 - Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_006'**

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_006
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with an invalid signature (signed body changed after signature calculation), current SessionID, OEMProvisioning certificate and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_006	

Table 296 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_006'.

Table **297 - Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_007'**

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_007
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with a valid signature, current SessionID, invalid OEMProvisioning certificate (Expired) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with the current SessionID, ResponseCode 'FAILED_CertificateExpired' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-752], [V2G2-468], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_007	

Table 297 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **298 - Test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_008'**

TC Id	TC_SECC_CMN_VTB_CertificateInstallation_008
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationReq message with a valid signature, current SessionID, invalid OEMProvisioning certificate (Invalid content -> Domain component changed) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateInstallationRes message with the current SessionID, ResponseCode 'FAILED' or 'FAILED_NoCertificateAvailable' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-237], [V2G2-238], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateInstallation_007	

Table 298 lists the test case description for 'TC_SECC_CMN_VTB_CertificateInstallation_008'.

10.2.3.13 SECC test cases for CertificateUpdate

10.2.3.13.1 General information

The following sub clauses include all *V2G CertificateUpdate message* test case descriptions, where the SUT is the SECC.

10.2.3.13.2 Common test cases

Table **299 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_001'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_001
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.10.3; 8.7.3.1; 8.8.4.3; 8.5.2.28; 8.5.2.29; 8.5.2.30; 8.6.2.3.2; 8.6.2.4.2; 8.6.3.4; 8.8.3.1; 8.6.2.1; 8.7.4.4; 7.9.2.4.5; 8.5.2.26; 7.9.2.4.3; 7.3.2; 7.3.3; AnnexF; 7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-274], [V2G2-275], [V2G2-557], [V2G2-752], [V2G2-778], [V2G2-779], [V2G2-780], [V2G2-781], [V2G2-782], [V2G2-783], [V2G2-389], [V2G2-401], [V2G2-412], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-653], [V2G2-353], [V2G2-354], [V2G2-814], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-884], [V2G2-822], [V2G2-665], [V2G2-103], [V2G2-104]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_001	

Table 299 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **300 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_002'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a CertificateUpdateReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.3.3; 8.7.3.1; 8.8.3.1; 8.8.4.3; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-557], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_002	

Table 300 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_002'.

Table **301 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_003'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_003
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, invalid SessionID, contract certificate chain, valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.10.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_003	

Table 301 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **302 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_004'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_004	

Table 302 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_004'.

Table **303 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_005'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_005
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with an invalid signature (wrong private key used), current SessionID, valid EMAID contract certificate chain, valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_005	

Table 303 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **304 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_006'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_006
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with an invalid signature (signed body changed after signature calculation), current SessionID, contract certificate chain, valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-461], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_006	

Table 304 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_006'.

Table **305 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_008'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_008
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Leaf Certificate from wrong PKI is selected), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_CertChainError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-470], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_007	

Table 305 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **306 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_009'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_009
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Sub2 Certificate from wrong PKI is selected), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_CertChainError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-470], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_007	

Table 306 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_009'.

Table **307 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_010'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_010
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Sub1 Certificate from wrong PKI is selected), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_CertChainError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-470], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_007	

Table 307 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **308 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_011'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_011
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Certificate chain from wrong PKI is selected), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_CertChainError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-470], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_007	

Table 308 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_011'.

Table **309 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_012'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_012
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, wrong EMAID (BodyType CertificateUpdateReq) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED', 'FAILED_NoCertificateAvailable' or 'FAILED_ContractCanceled' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_008	

Table 309 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **310 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_013'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_013
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, invalid EMAID (BodyType CertificateUpdateReq) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED', 'FAILED_NoCertificateAvailable' or 'FAILED_ContractCanceled' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'..
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_008	

Table 310 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_013'.

Table **311 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_014'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_014
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, wrong EMAID (contract certificate) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED', 'FAILED_NoCertificateAvailable' or 'FAILED_ContractCanceled' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_009	

Table 311 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **312 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_015'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_015
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, invalid EMAID (contract certificate) and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED', 'FAILED_NoCertificateAvailable' or 'FAILED_ContractCanceled' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_009	

Table 312 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_015'.

Table **313 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_016'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_016
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Expired), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED_CertificateExpired' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-473], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_010	

Table 313 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **314 - Test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_017'**

TC Id	TC_SECC_CMN_VTB_CertificateUpdate_017
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateReq message with a valid signature, current SessionID, invalid contract certificate chain (Invalid content -> Country name was set), valid EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a CertificateUpdateRes message with the current SessionID, ResponseCode 'FAILED' or 'FAILED_NoCertificateAvailable' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.11.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-231], [V2G2-232], [V2G2-752], [V2G2-458], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-696]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_SECC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_CMN_PR_PaymentServiceSelection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_CertificateUpdate_010	

Table 314 lists the test case description for 'TC_SECC_CMN_VTB_CertificateUpdate_017'.

10.2.3.14 SECC test cases for SessionStop

10.2.3.14.1 General information

The following sub clauses include all *V2G SessionStop message* test case descriptions, where the SUT is the SECC.

10.2.3.14.2 AC specific test cases

Table **315 - Test case description for 'TC_SECC_AC_VTB_SessionStop_001'**

TC Id	TC_SECC_AC_VTB_SessionStop_001
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the data link connection within 'par_T_match_leave'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.3; 8.8.4.3.1; 8.6.2.3.1; 8.6.2.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-571], [V2G2-751], [V2G2-752], [V2G2-377], [V2G2-381], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-724], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_001	

Table 315 lists the test case description for 'TC_SECC_AC_VTB_SessionStop_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **316 - Test case description for 'TC_SECC_AC_VTB_SessionStop_002'**

TC Id	TC_SECC_AC_VTB_SessionStop_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a SessionStopReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.3; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_002	

Table 316 lists the test case description for 'TC_SECC_AC_VTB_SessionStop_002'.

Table **317 - Test case description for 'TC_SECC_AC_VTB_SessionStop_003'**

TC Id	TC_SECC_AC_VTB_SessionStop_003
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with an invalid SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.3; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_003	

Table 317 lists the test case description for 'TC_SECC_AC_VTB_SessionStop_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **318 - Test case description for 'TC_SECC_AC_VTB_SessionStop_004'**

TC Id	TC_SECC_AC_VTB_SessionStop_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_004	

Table 318 lists the test case description for 'TC_SECC_AC_VTB_SessionStop_004'.

10.2.3.14.3 DC specific test cases

Table **319 - Test case description for 'TC_SECC_DC_VTB_SessionStop_001'**

TC Id	TC_SECC_DC_VTB_SessionStop_001
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the data link connection within 'par_T_match_leave'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.3; 8.8.4.3.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-571], [V2G2-751], [V2G2-752], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-568], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-921], [V2G2-724], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := false
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_001	

Table 319 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **320 - Test case description for 'TC_SECC_DC_VTB_SessionStop_002'**

TC Id	TC_SECC_DC_VTB_SessionStop_002
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the data link connection within 'par_T_match_leave'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.3; 8.8.4.3.1; 8.6.2.4.1; 8.6.2.4.2; 7.3.4; 8.6.2.1; 8.7.4.4; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-571], [V2G2-751], [V2G2-752], [V2G2-391], [V2G2-395], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-921], [V2G2-724], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_WeldingDetection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_001	

Table 320 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_002'.

Table **321 - Test case description for 'TC_SECC_DC_VTB_SessionStop_003'**

TC Id	TC_SECC_DC_VTB_SessionStop_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a SessionStopReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.3; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := false
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_002	

Table 321 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **322 - Test case description for 'TC_SECC_DC_VTB_SessionStop_004'**

TC Id	TC_SECC_DC_VTB_SessionStop_004
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with an invalid SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.3; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := false
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_003	

Table 322 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_004'.

Table **323 - Test case description for 'TC_SECC_DC_VTB_SessionStop_005'**

TC Id	TC_SECC_DC_VTB_SessionStop_005
Test objective	Test System executes GoodCase procedure and stops the charging process by sending a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. After receipt of PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters, Test System waits until the 'V2G_Sequence_Timeout' timer has expired. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := false
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_004	

Table 323 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **324 - Test case description for 'TC_SECC_DC_VTB_SessionStop_006'**

TC Id	TC_SECC_DC_VTB_SessionStop_006
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters without signaling CP State B before. Test System then checks that the SUT sends a SessionStopRes message with the current SessionID, ResponseCode 'FAILED' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.3; 8.8.3.1; 8.8.4.3.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-752], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-458], [V2G2-922], [V2G2-919], [V2G2-920]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := false
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_005	

Table 324 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_006'.

Table **325 - Test case description for 'TC_SECC_DC_VTB_SessionStop_007'**

TC Id	TC_SECC_DC_VTB_SessionStop_007
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a SessionStopReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.3; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_WeldingDetection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_002	

Table 325 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **326 - Test case description for 'TC_SECC_DC_VTB_SessionStop_008'**

TC Id	TC_SECC_DC_VTB_SessionStop_008
Test objective	Test System executes GoodCase procedure and sends a SessionStopReq message with an invalid SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Test System then checks that the SUT sends a SessionStopRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.3; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-240], [V2G2-241], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_WeldingDetection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_003	

Table 326 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_008'.

Table **327 - Test case description for 'TC_SECC_DC_VTB_SessionStop_009'**

TC Id	TC_SECC_DC_VTB_SessionStop_009
Test objective	Test System executes GoodCase procedure with welding detection process and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_WeldingDetection_001	
Expected behavior	
f_SECC_CMN_TB_VTB_SessionStop_004	

Table 327 lists the test case description for 'TC_SECC_DC_VTB_SessionStop_009'.

10.2.3.15 SECC test cases for ChargingStatus

10.2.3.15.1 General information

The following sub clause includes all *V2G ChargingStatus message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.15.2 AC specific test cases

Table **328 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_001'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_001
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.4.2.3, 8.6.3.2; 8.8.4.3.2; 8.5.2.7; 8.5.3.1; 8.6.2.3.2; 8.6.3.2; 8.8.3.1; 7.3.4; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-407], [V2G2-691], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := none_, PIXIT_SECC_CMN_MeterInfo := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_001	

Table 328 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_001'.

Table **329 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_002'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_002
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.4.2.3, 8.6.3.2; 8.8.4.3.2; 8.5.2.7; 8.5.3.1; 8.6.2.3.2; 8.6.3.2; 8.8.3.1; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-406], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_001	

Table 329 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **330 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_003'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a ChargingStatusReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.3, 8.6.3.2; 8.8.3.1; 8.8.4.3.2; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_002	

Table 330 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_003'.

Table **331 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_004'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_004
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with an invalid SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.4.2.3, 8.6.3.2; 8.8.3.1; 8.8.4.3.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_003	

Table 331 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **332 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_005'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_004	

Table 332 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_005'.

Table **333 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_006'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_006
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false', a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.4.2.3, 8.6.3.2; 8.8.4.3.2; 8.5.2.7; 8.5.3.1; 8.6.2.3.2; 8.6.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-407], [V2G2-691], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665], [V2G2-276], [V2G2-277]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := none_, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_005	

Table 333 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **334 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_007'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_007
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true', a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.4.2.3, 8.6.3.2; 8.8.4.3.2; 8.5.2.7; 8.5.3.1; 8.6.2.3.2; 8.6.3.2; 8.8.3.1; 8.6.2.1; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-406], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665], [V2G2-276], [V2G2-277]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_005	

Table 334 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_007'.

Table **335 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_008'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_008
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false', a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.4.2.3, 8.6.3.2; 8.8.4.3.2; 8.5.2.7; 8.5.3.1; 8.6.2.3.2; 8.6.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-407], [V2G2-691], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665], [V2G2-276], [V2G2-277], [V2G2-831]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := none_, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := meterReading
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_005	

Table 335 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **336 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_009'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_009
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true', a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.4.2.3, 8.6.3.2; 8.8.4.3.2; 8.5.2.7; 8.5.3.1; 8.6.2.3.2; 8.6.3.2; 8.8.3.1; 8.6.2.1; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-406], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665], [V2G2-276], [V2G2-277], [V2G2-831]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := meterReading
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_005	

Table 336 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_009'.

Table **337 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_010'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_010
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.4.2.3, 8.6.3.2; 8.8.4.3.2; 8.5.2.7; 8.5.3.1; 8.6.2.3.2; 8.8.3.1; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_001	

Table 337 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **338 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_011'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_011
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.4.2.3, 8.6.3.2; 8.8.4.3.2; 8.5.2.7; 8.5.3.1; 8.6.2.3.2; 8.8.3.1; 8.6.2.1; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665], [V2G2-276], [V2G2-277]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := unknown
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_005	

Table 338 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_011'.

Table **339 - Test case description for 'TC_SECC_AC_VTB_ChargingStatus_012'**

TC Id	TC_SECC_AC_VTB_ChargingStatus_012
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.4.2.3, 8.6.3.2; 8.8.4.3.2; 8.5.2.7; 8.5.3.1; 8.6.2.3.2; 8.8.3.1; 8.6.2.1; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-243], [V2G2-244], [V2G2-575], [V2G2-278], [V2G2-279], [V2G2-752], [V2G2-577], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-665], [V2G2-276], [V2G2-277], [V2G2-831]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := meterReading
PreCondition	
f_SECC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_AC_TB_VTB_ChargingStatus_005	

Table 339 lists the test case description for 'TC_SECC_AC_VTB_ChargingStatus_012'.

10.2.3.16 SECC test cases for MeteringReceipt

10.2.3.16.1 General information

The following sub clauses include all *V2G MeteringReceipt message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.16.2 AC specific test cases

Table **340 - Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_001'**

TC Id	TC_SECC_AC_VTB_MeteringReceipt_001
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.3; 8.8.4.3.2; 8.4.2; 8.5.3.1; 8.6.2.3.2; 8.8.3.1; 8.6.2.1; 7.3.2; 7.9.2.4.5; 8.4.3.13.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-580], [V2G2-752], [V2G2-358], [V2G2-359], [V2G2-381], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-665], [V2G2-904]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_AC_TB_VTB_MeteringReceipt_001	

Table 340 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_001'.

Table **341 - Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_002'**

TC Id	TC_SECC_AC_VTB_MeteringReceipt_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a MeteringReceiptReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.3.3; 8.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_CMN_TB_VTB_MeteringReceipt_001	

Table 341 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **342 - Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_003'**

TC Id	TC_SECC_AC_VTB_MeteringReceipt_003
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, invalid sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.3; 8.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_AC_TB_VTB_MeteringReceipt_002	

Table 342 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_003'.

Table **343 - Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_004'**

TC Id	TC_SECC_AC_VTB_MeteringReceipt_004
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, invalid MeterReading and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.3; 8.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_AC_TB_VTB_MeteringReceipt_003	

Table 343 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **344 - Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_005'**

TC Id	TC_SECC_AC_VTB_MeteringReceipt_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_CMN_TB_VTB_MeteringReceipt_002	

Table 344 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_005'.

Table **345 - Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_006'**

TC Id	TC_SECC_AC_VTB_MeteringReceipt_006
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with an invalid signature (wrong private key used), current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.3; 8.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_AC_TB_VTB_MeteringReceipt_004	

Table 345 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **346 - Test case description for 'TC_SECC_AC_VTB_MeteringReceipt_007'**

TC Id	TC_SECC_AC_VTB_MeteringReceipt_007
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with an invalid signature (signed body changed after signature calculation), current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.3; 8.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_AC_PR_ChargingStatus_001	
Expected behavior	
f_SECC_AC_TB_VTB_MeteringReceipt_005	

Table 346 lists the test case description for 'TC_SECC_AC_VTB_MeteringReceipt_007'.

10.2.3.16.3 DC specific test cases

Table **347 - Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_001'**

TC Id	TC_SECC_DC_VTB_MeteringReceipt_001
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.3; 8.8.4.3.3; 8.4.2; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4; 7.3.2; 7.9.2.4.5; 8.4.3.13.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-797], [V2G2-752], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-005], [V2G2-007], [V2G2-653], [V2G2-665], [V2G2-904]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_DC_TB_VTB_MeteringReceipt_001	

Table 347 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **348 - Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_002'**

TC Id	TC_SECC_DC_VTB_MeteringReceipt_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a MeteringReceiptReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.3.3; 8.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_CMN_TB_VTB_MeteringReceipt_001	

Table 348 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_002'.

Table **349 - Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_003'**

TC Id	TC_SECC_DC_VTB_MeteringReceipt_003
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, invalid sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.3; 8.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_DC_TB_VTB_MeteringReceipt_002	

Table 349 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **350 - Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_004'**

TC Id	TC_SECC_DC_VTB_MeteringReceipt_004
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, invalid MeterReading and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.3; 8.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_DC_TB_VTB_MeteringReceipt_003	

Table 350 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_004'.

Table **351 - Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_005'**

TC Id	TC_SECC_DC_VTB_MeteringReceipt_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false' and current and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_CMN_TB_VTB_MeteringReceipt_002	

Table 351 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **352 - Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_006'**

TC Id	TC_SECC_DC_VTB_MeteringReceipt_006
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with an invalid signature (wrong private key used), current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.3; 8.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_DC_TB_VTB_MeteringReceipt_004	

Table 352 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_006'.

Table **353 - Test case description for 'TC_SECC_DC_VTB_MeteringReceipt_007'**

TC Id	TC_SECC_DC_VTB_MeteringReceipt_007
Test objective	Test System executes GoodCase procedure and sends a MeteringReceiptReq message with an invalid signature (signed body changed after signature calculation), current sessionID, valid SAScheduleTupleId, metering information and all additional mandatory parameters. Test System then checks that the SUT sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.3; 8.4.2; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-247], [V2G2-248], [V2G2-752], [V2G2-481], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_SECC_DC_PR_CurrentDemand_001	
Expected behavior	
f_SECC_DC_TB_VTB_MeteringReceipt_005	

Table 353 lists the test case description for 'TC_SECC_DC_VTB_MeteringReceipt_007'.

10.2.3.17 SECC test cases for CableCheck

10.2.3.17.1 General information

The following sub clause includes all *V2G CableCheck message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

10.2.3.17.2 DC specific test cases

Table **354 - Test case description for 'TC_SECC_DC_VTB_CableCheck_001'**

TC Id	TC_SECC_DC_VTB_CableCheck_001
Test objective	Test System executes GoodCase procedure, signals CP State C and sends a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a CableCheckRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another CableCheckReq message. This message sequence will be repeated until the cable check of the SUT has finished or 'V2G_EVCC_CableCheck_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.5.2.3; 8.5.4.1; 8.7.3.1; 8.8.4.3; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-584], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-917], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_EvStateInChargeProgress := e_StateC
PIXIT selection	PIXIT_SECC_DC_CableCheckOngoing == unknown
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_001	

Table 354 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_001'.

Table **355 - Test case description for 'TC_SECC_DC_VTB_CableCheck_002'**

TC Id	TC_SECC_DC_VTB_CableCheck_002
Test objective	Test System executes GoodCase procedure, signals CP State D and sends a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. If a CableCheckRes message with EVSEProcessing 'Ongoing' was received before, the Test System sends another CableCheckReq message. This message sequence will be repeated until the cable check of the SUT has finished or 'V2G_EVCC_CableCheck_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.5.2.3; 8.5.4.1; 8.7.3.1; 8.8.4.3; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-364], [V2G2-365], [V2G2-584], [V2G2-752], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-917], [V2G2-915], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_EvStateInChargeProgress := e_StateD
PIXIT selection	PIXIT_SECC_DC_CableCheckOngoing == unknown
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_001	

Table 355 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **356 - Test case description for 'TC_SECC_DC_VTB_CableCheck_003'**

TC Id	TC_SECC_DC_VTB_CableCheck_003
Test objective	Test System executes GoodCase procedure, signals CP State C or D and sends a valid ServiceDiscoveryReq message instead of a CableCheckReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.4.3.3.3; 8.7.3.1; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_002	

Table 356 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_003'.

Table **357 - Test case description for 'TC_SECC_DC_VTB_CableCheck_004'**

TC Id	TC_SECC_DC_VTB_CableCheck_004
Test objective	Test System executes GoodCase procedure, signals CP State C or D and sends a CableCheckReq message with an invalid SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.4.5.2.3; 8.7.3.1; 8.8.3.1; 8.8.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_003	

Table 357 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **358 - Test case description for 'TC_SECC_DC_VTB_CableCheck_005'**

TC Id	TC_SECC_DC_VTB_CableCheck_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_004	

Table 358 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_005'.

Table **359 - Test case description for 'TC_SECC_DC_VTB_CableCheck_006'**

TC Id	TC_SECC_DC_VTB_CableCheck_006
Test objective	Test System executes GoodCase procedure, signals CP State C and sends a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Furthermore the repetition of the CableCheck message sequence will be checked until the cable check of the SUT has finished or 'V2G_EVCC_CableCheck_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.5.2.3; 8.5.4.1; 8.7.3.1; 8.8.4.3; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4; 8.8.4.3.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-364], [V2G2-365], [V2G2-584], [V2G2-752], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-917], [V2G2-915], [V2G2-665], [V2G2-621]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_EvStateInChargeProgress := e_StateC
PIXIT selection	PIXIT_SECC_DC_CableCheckOngoing == ongoing
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_001	

Table 359 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **360 - Test case description for 'TC_SECC_DC_VTB_CableCheck_007'**

TC Id	TC_SECC_DC_VTB_CableCheck_007
Test objective	Test System executes GoodCase procedure, signals CP State D and sends a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Furthermore the repetition of the CableCheck message sequence will be checked until the cable check of the SUT has finished or 'V2G_EVCC_CableCheck_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.5.2.3; 8.5.4.1; 8.7.3.1; 8.8.4.3; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4; 8.8.4.3.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-364], [V2G2-365], [V2G2-584], [V2G2-752], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-917], [V2G2-915], [V2G2-665], [V2G2-621]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_CMN_EvStateInChargeProgress := e_StateD
PIXIT selection	PIXIT_SECC_DC_CableCheckOngoing == ongoing
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_001	

Table 360 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_007'.

Table **361 - Test case description for 'TC_SECC_DC_VTB_CableCheck_008'**

TC Id	TC_SECC_DC_VTB_CableCheck_008
Test objective	Test System executes GoodCase procedure, signals no CP State C or D and sends a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a CableCheckRes message with the current SessionID, ResponseCode 'FAILED' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.4.5.2.3; 8.7.3.1; 8.8.3.1; 8.8.4.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-251], [V2G2-252], [V2G2-752], [V2G2-918], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_SECC_DC_TB_VTB_CableCheck_005	

Table 361 lists the test case description for 'TC_SECC_DC_VTB_CableCheck_008'.

10.2.3.18 SECC test cases for PreCharge

10.2.3.18.1 General information

The following sub clause includes all *V2G PreCharge message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.18.2 DC specific test cases

Table **362 - Test case description for 'TC_SECC_DC_VTB_PreCharge_001'**

TC Id	TC_SECC_DC_VTB_PreCharge_001
Test objective	Test System executes GoodCase procedure and sends a PreChargeReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a PreChargeRes message with the current SessionID, ResponseCode 'OK', EVSEStatusCode 'EVSE_Ready' and all additional mandatory parameters. If the Test System sends another valid PreChargeReq message, the SUT shall respond to this request message again. This message sequence will be repeated until the required voltage has been reached or 'V2G_EVCC_PreCharge_Timer' has expired.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.5.3; 8.7.3.7; 8.5.2.7; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-255], [V2G2-256], [V2G2-278], [V2G2-279], [V2G2-587], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CableCheck_001	
Expected behavior	
f_SECC_DC_TB_VTB_PreCharge_001	

Table 362 lists the test case description for 'TC_SECC_DC_VTB_PreCharge_001'.

Table **363 - Test case description for 'TC_SECC_DC_VTB_PreCharge_002'**

TC Id	TC_SECC_DC_VTB_PreCharge_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a PreChargeReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.3.3.3; 8.7.3.7; 8.5.4.1; 8.8.3.1; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CableCheck_001	
Expected behavior	
f_SECC_DC_TB_VTB_PreCharge_002	

Table 363 lists the test case description for 'TC_SECC_DC_VTB_PreCharge_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **364 - Test case description for 'TC_SECC_DC_VTB_PreCharge_003'**

TC Id	TC_SECC_DC_VTB_PreCharge_003
Test objective	Test System executes GoodCase procedure and sends a PreChargeReq message with an invalid SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a PreChargeRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 15118-2.7.4; 8.4.5.3; 8.7.3.7; 8.5.4.1; 8.8.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-255], [V2G2-256], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CableCheck_001	
Expected behavior	
f_SECC_DC_TB_VTB_PreCharge_003	

Table 364 lists the test case description for 'TC_SECC_DC_VTB_PreCharge_003'.

Table **365 - Test case description for 'TC_SECC_DC_VTB_PreCharge_004'**

TC Id	TC_SECC_DC_VTB_PreCharge_004
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_CableCheck_001	
Expected behavior	
f_SECC_DC_TB_VTB_PreCharge_004	

Table 365 lists the test case description for 'TC_SECC_DC_VTB_PreCharge_004'.

10.2.3.19 SECC test cases for CurrentDemand

10.2.3.19.1 General information

The following sub clause includes all *V2G CurrentDemand message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.19.2 DC specific test cases

Table **366 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_001'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_001
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.4.3; 8.5.4.1; 8.8.4.3.3; 8.5.2.7; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.2; 8.6.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-788], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := none_, PIXIT_SECC_CMN_MeterInfo := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_001	

Table 366 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_001'.

Table **367 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_002'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_002
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.4.3; 8.5.4.1; 8.8.4.3.3; 8.5.2.7; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.2; 8.8.3.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-395], [V2G2-787], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-795], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_001	

Table 367 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **368 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_003'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_003
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a CurrentDemandReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.3; 8.5.4.1; 8.8.3.1; 8.8.4.3.3; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_002	

Table 368 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_003'.

Table **369 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_004'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_004
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with an unknown SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.4.3; 8.8.3.1; 8.8.4.3.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-259], [V2G2-260], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_003	

Table 369 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **370 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_005'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_005
Test objective	Test System executes GoodCase procedure and waits until the 'V2G_Sequence_Timeout' timer has expired after receiving a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_004	

Table 370 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_005'.

Table **371 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_006'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_006
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false', a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.4.3; 8.5.4.1; 8.8.4.3.3; 8.5.2.7; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.2; 8.6.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-788], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-276], [V2G2-277]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := none_, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_005	

Table 371 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **372 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_007'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_007
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true', a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.4.3; 8.5.4.1; 8.8.4.3.3; 8.5.2.7; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.2; 8.8.3.1; 8.6.2.1; 8.7.4.4; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-395], [V2G2-787], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-795], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-276], [V2G2-277]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_005	

Table 372 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_007'.

Table **373 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_008'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_008
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'false', a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.4.3; 8.5.4.1; 8.8.4.3.3; 8.5.2.7; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.2; 8.6.3.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.7.4.4; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-788], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-276], [V2G2-277], [V2G2-831]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := none_, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := meterReading
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_005	

Table 373 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **374 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_009'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_009
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true', a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.4.3; 8.5.4.1; 8.8.4.3.3; 8.5.2.7; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.2; 8.8.3.1; 8.6.2.1; 8.7.4.4; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-395], [V2G2-787], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-795], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-276], [V2G2-277], [V2G2-831]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := receiptRequired, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := meterReading
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_005	

Table 374 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_009'.

Table **375 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_010'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_010
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.4.3; 8.5.4.1; 8.8.4.3.3; 8.5.2.7; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-795], [V2G2-915], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_001	

Table 375 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **376 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_011'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_011
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.4.3; 8.5.4.1; 8.8.4.3.3; 8.5.2.7; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 8.6.2.1; 8.7.4.4; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-795], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-276], [V2G2-277]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := unknown
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_005	

Table 376 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_011'.

Table **377 - Test case description for 'TC_SECC_DC_VTB_CurrentDemand_012'**

TC Id	TC_SECC_DC_VTB_CurrentDemand_012
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.4.3; 8.5.4.1; 8.8.4.3.3; 8.5.2.7; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.2; 8.8.3.1; 8.6.2.1; 8.7.4.4; 8.5.2.6
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-278], [V2G2-279], [V2G2-259], [V2G2-260], [V2G2-593], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-395], [V2G2-457], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-795], [V2G2-915], [V2G2-832], [V2G2-665], [V2G2-276], [V2G2-277], [V2G2-831]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_SECC_CMN_ReceiptRequired := unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_SECC_CMN_MeterInfo := meterInfo, PIXIT_CMN_CMN_MeterReading := meterReading
PreCondition	
f_SECC_DC_PR_PowerDeliveryStart_001	
Expected behavior	
f_SECC_DC_TB_VTB_CurrentDemand_005	

Table 377 lists the test case description for 'TC_SECC_DC_VTB_CurrentDemand_012'.

10.2.3.20 SECC test cases for WeldingDetection

10.2.3.20.1 General information

The following sub clause includes all *V2G WeldingDetection message* test case descriptions, where the SUT is the SECC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.2.3.20.2 DC specific test cases

Table **378 - Test case description for 'TC_SECC_DC_VTB_WeldingDetection_001'**

TC Id	TC_SECC_DC_VTB_WeldingDetection_001
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.5.3; 8.8.4.3.3; 8.5.2.7; 8.5.4.1; 8.6.2.4.1; 8.6.2.4.2; 8.8.3.1; 7.3.4; 8.6.2.1; 8.8.4.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-263], [V2G2-264], [V2G2-278], [V2G2-279], [V2G2-597], [V2G2-752], [V2G2-364], [V2G2-365], [V2G2-366], [V2G2-391], [V2G2-395], [V2G2-457], [V2G2-634], [V2G2-664], [V2G2-665], [V2G2-666], [V2G2-667], [V2G2-677], [V2G2-915], [V2G2-921], [V2G2-832], [V2G2-665]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_WeldingDetection_001	

Table 378 lists the test case description for 'TC_SECC_DC_VTB_WeldingDetection_001'.

Table **379 - Test case description for 'TC_SECC_DC_VTB_WeldingDetection_002'**

TC Id	TC_SECC_DC_VTB_WeldingDetection_002
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryReq message instead of a WeldingDetectionReq message. Test System then checks that the SUT sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.3; 8.8.3.1; 8.8.4.3.3; 8.8.4.3.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-195], [V2G2-196], [V2G2-752], [V2G2-459], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-538]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_WeldingDetection_002	

Table 379 lists the test case description for 'TC_SECC_DC_VTB_WeldingDetection_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **380 - Test case description for 'TC_SECC_DC_VTB_WeldingDetection_003'**

TC Id	TC_SECC_DC_VTB_WeldingDetection_003
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionReq message with an invalid SessionID and all additional mandatory parameters. Test System then checks that the SUT sends a WeldingDetectionRes message with a valid SessionID, ResponseCode 'FAILED_UnknownSession' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.5.3; 8.8.3.1; 8.8.4.3.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-263], [V2G2-264], [V2G2-752], [V2G2-460], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_WeldingDetection_003	

Table 380 lists the test case description for 'TC_SECC_DC_VTB_WeldingDetection_003'.

Table **381 - Test case description for 'TC_SECC_DC_VTB_WeldingDetection_004'**

TC Id	TC_SECC_DC_VTB_WeldingDetection_004
Test objective	Test System executes GoodCase procedure and stops the charging process by sending a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. After receipt of PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters, Test System waits until the 'V2G_Sequence_Timeout' timer has expired. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.8.4.3.1; 8.7.2.1; 8.7.2.3
Referenced requirement(s)	[V2G2-729], [V2G2-034], [V2G2-435], [V2G2-441], [V2G2-442], [V2G2-443]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_WeldingDetection_004	

Table 381 lists the test case description for 'TC_SECC_DC_VTB_WeldingDetection_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **382 - Test case description for 'TC_SECC_DC_VTB_WeldingDetection_005'**

TC Id	TC_SECC_DC_VTB_WeldingDetection_005
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionReq message with the current SessionID and all additional mandatory parameters without signaling CP State B before. Test System then checks that the SUT sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'FAILED' and only additional mandatory parameters without optional parameters. The test system then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 15118-2.7.4; 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.5.5.3; 8.8.3.1; 8.8.4.3.3; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-263], [V2G2-264], [V2G2-752], [V2G2-539], [V2G2-729], [V2G2-736], [V2G2-034], [V2G2-458], [V2G2-922], [V2G2-919], [V2G2-920]
Config Id	CF_04_001
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_DC_PR_PowerDeliveryStop_001	
Expected behavior	
f_SECC_DC_TB_VTB_WeldingDetection_005	

Table 382 lists the test case description for 'TC_SECC_DC_VTB_WeldingDetection_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3 EVCC test cases

10.3.1 General information

This sub clause covers all V2G Application layer test case descriptions, where the SUT is the EVCC.

10.3.2 V2G protocol handshake

10.3.2.1 General information

This sub clause covers the *V2G protocol handshake* test case descriptions, where the SUT is the EVCC.

10.3.2.2 EVCC test cases for SupportedAppProtocol

Table **383 - Test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_001'**

TC Id	TC_EVCC_CMN_VTB_SupportedAppProtocol_001
Test objective	Test System executes GoodCase procedure and sends an SDP response message with 'transportProtocol' equals to '0x00'H and a valid security option, port and IP address. Test System then checks that the SUT sends a correct SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.2.1; 8.2.2; 8.2.3; 8.8.4.2.1; 7.4; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.4.1; 8.6.2.4.2; 7.10.1.7; 8.6.2.1; 7.7.3.4; 7.3.3; 7.7.3.3
Referenced requirement(s)	[V2G2-165], [V2G2-166], [V2G2-167], [V2G2-175], [V2G2-178], [V2G2-483], [V2G2-022], [V2G2-024], [V2G2-376], [V2G2-380], [V2G2-390], [V2G2-394], [V2G2-628], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-021], [V2G2-603], [V2G2-645], [V2G2-008], [V2G2-875]
Config Id	CF_04_002
PICS selection	
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_SDP_001	
Expected behavior	
f_EVCC_CMN_PR_InitConfiguration_002, f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001	

Table 383 lists the test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_001'.

Table **384 - Test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_002'**

TC Id	TC_EVCC_CMN_VTB_SupportedAppProtocol_002
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'FAILED_NoNegotiation' and all additional mandatory parameters. Test System then checks that the SUT closes the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.2.1
Referenced requirement(s)	[V2G2-173], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SupportedAppProtocol_002	

Table 384 lists the test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **385 - Test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_003'**

TC Id	TC_EVCC_CMN_VTB_SupportedAppProtocol_003
Test objective	Test System executes GoodCase procedure, sends a valid SDP response message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters. Test System then checks that the SUT closes the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-484], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SupportedAppProtocol_003	

Table 385 lists the test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_003'.

Table **386 - Test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_004'**

TC Id	TC_EVCC_CMN_VTB_SupportedAppProtocol_004
Test objective	Test System executes GoodCase procedure and sends an SDP response message with 'transportProtocol' equals to '0x00'H and a valid security option, port and IP address. Test System then checks that the SUT sends a correct SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.2.1; 8.2.2; 8.2.3; 8.8.4.2.1; 7.4; 8.6.2.3.1; 8.6.2.4.1; 7.10.1.7; 8.6.2.1
Referenced requirement(s)	[V2G2-165], [V2G2-166], [V2G2-167], [V2G2-175], [V2G2-178], [V2G2-483], [V2G2-022], [V2G2-024], [V2G2-376], [V2G2-390], [V2G2-628], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-645], [V2G2-646]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_CMN_PR_SDP_001	
Expected behavior	
f_EVCC_CMN_PR_InitConfiguration_002, f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001	

Table 386 lists the test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **387 - Test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_005'**

TC Id	TC_EVCC_CMN_VTB_SupportedAppProtocol_005
Test objective	Test System executes GoodCase procedure and sends an SDP response message with 'transportProtocol' equals to '0x00'H and a valid security option, port and IP address. Test System then checks that the SUT sends a correct SupportedAppProtocolReq message with a list of valid AppProtocols including ISO namespace and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.2.1; 8.2.2; 8.2.3; 8.8.4.2.1; 7.4; 8.6.2.3.1; 8.6.2.4.1; 7.10.1.7; 8.6.2.1
Referenced requirement(s)	[V2G2-165], [V2G2-166], [V2G2-167], [V2G2-175], [V2G2-178], [V2G2-483], [V2G2-022], [V2G2-024], [V2G2-376], [V2G2-390], [V2G2-628], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-645]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_CMN_PR_SDP_001	
Expected behavior	
f_EVCC_CMN_PR_InitConfiguration_002, f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001	

Table 387 lists the test case description for 'TC_EVCC_CMN_VTB_SupportedAppProtocol_005'.

10.3.3 V2G messages

10.3.3.1 General information

This sub clause includes all *V2G message* test case descriptions, where the SUT is the EVCC.

10.3.3.2 EVCC test cases for SessionSetup

10.3.3.3 General information

The following sub clauses include all *V2G SessionSetup message* test case descriptions, where the SUT is the EVCC.

10.3.3.3.1 Common test cases

Table **388 - Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_001'**

TC Id	TC_EVCC_CMN_VTB_SessionSetup_001
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with SessionID '0', EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.2; 8.4.3.2.1; 8.4.3.2.2; 8.8.4.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-746], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_001	

Table 388 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **389 - Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_002'**

TC Id	TC_EVCC_CMN_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionSetupRes message with a SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-486], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_002	

Table 389 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_002'.

Table **390 - Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_003'**

TC Id	TC_EVCC_CMN_VTB_SessionSetup_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionSetupRes message with a SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-486], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_002	

Table 390 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **391 - Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_004'**

TC Id	TC_EVCC_CMN_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure, sends a valid SupportedAppProtocolRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a SessionSetupReq message with SessionID '0', valid EvccID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-486], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_003	

Table 391 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_004'.

Table **392 - Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_005'**

TC Id	TC_EVCC_CMN_VTB_SessionSetup_005
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of SessionSetupRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_004	

Table 392 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **393 - Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_006'**

TC Id	TC_EVCC_CMN_VTB_SessionSetup_006
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a SessionSetupRes with a new SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_005	

Table 393 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_006'.

Table **394 - Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_007'**

TC Id	TC_EVCC_CMN_VTB_SessionSetup_007
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a SessionSetupRes with a new SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_005	

Table 394 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **395 - Test case description for 'TC_EVCC_CMN_VTB_SessionSetup_008'**

TC Id	TC_EVCC_CMN_VTB_SessionSetup_008
Test objective	Test System executes GoodCase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionSetupReq message with SessionID '0', EvccID equals to the MAC adress of the SUT and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.2; 8.4.3.2.1; 8.4.3.2.2; 8.8.4.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-746], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-879]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SupportedAppProtocol_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_001	

Table 395 lists the test case description for 'TC_EVCC_CMN_VTB_SessionSetup_008'.

10.3.3.3.2 AC specific test cases

Table **396 - Test case description for 'TC_EVCC_AC_VTB_SessionSetup_001'**

TC Id	TC_EVCC_AC_VTB_SessionSetup_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.2; 8.4.3.2.1; 8.4.3.2.2; 8.8.4.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 396 lists the test case description for 'TC_EVCC_AC_VTB_SessionSetup_001'.

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **397 - Test case description for 'TC_EVCC_AC_VTB_SessionSetup_002'**

TC Id	TC_EVCC_AC_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.2; 8.4.3.2.1; 8.4.3.2.2; 8.8.4.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 397 lists the test case description for 'TC_EVCC_AC_VTB_SessionSetup_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **398 - Test case description for 'TC_EVCC_AC_VTB_SessionSetup_003'**

TC Id	TC_EVCC_AC_VTB_SessionSetup_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.2; 8.4.3.2.1; 8.4.3.2.2; 8.8.4.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 398 lists the test case description for 'TC_EVCC_AC_VTB_SessionSetup_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **399 - Test case description for 'TC_EVCC_AC_VTB_SessionSetup_004'**

TC Id	TC_EVCC_AC_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.2; 8.4.3.2.1; 8.4.3.2.2; 8.8.4.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 399 lists the test case description for 'TC_EVCC_AC_VTB_SessionSetup_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.3.3 DC specific test cases

Table **400 - Test case description for 'TC_EVCC_DC_VTB_SessionSetup_001'**

TC Id	`TC_EVCC_DC_VTB_SessionSetup_001`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.2; 8.4.3.2.1; 8.4.3.2.2; 8.8.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_SessionSetup_006`	

Table 400 lists the test case description for 'TC_EVCC_DC_VTB_SessionSetup_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **401 - Test case description for 'TC_EVCC_DC_VTB_SessionSetup_002'**

TC Id	TC_EVCC_DC_VTB_SessionSetup_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.2; 8.4.3.2.1; 8.4.3.2.2; 8.8.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 401 lists the test case description for 'TC_EVCC_DC_VTB_SessionSetup_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **402 - Test case description for 'TC_EVCC_DC_VTB_SessionSetup_003'**

TC Id	TC_EVCC_DC_VTB_SessionSetup_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.2; 8.4.3.2.1; 8.4.3.2.2; 8.8.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 402 lists the test case description for 'TC_EVCC_DC_VTB_SessionSetup_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **403 - Test case description for 'TC_EVCC_DC_VTB_SessionSetup_004'**

TC Id	TC_EVCC_DC_VTB_SessionSetup_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System then executes Goodcase procedure and sends a SupportedAppProtocolRes message with ResponseCode 'OK_SuccessfulNegotiation', a valid schema and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct SessionSetupReq message with the previous SessionID, valid EvccID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.2; 8.4.3.2.1; 8.4.3.2.2; 8.8.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-188], [V2G2-189], [V2G2-485], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744], [V2G2-748]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_SessionSetup_006	

Table 403 lists the test case description for 'TC_EVCC_DC_VTB_SessionSetup_004'.

10.3.3.4 EVCC test cases for ServiceDiscovery

10.3.3.4.1 General information

The following sub clauses include all *V2G ServiceDiscovery message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.4.2 Common test cases

Table **404 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_001'**

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a SessionSetupRes message with a new SessionID, ResponseCode 'OK_NewSessionEstablished', valid EvseId and all additional mandatory parameters. Test System then checks that the SUT sends a correct ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.3.2; 8.4.3.3.3; 8.8.4.3.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-193], [V2G2-194], [V2G2-487], [V2G2-747], [V2G2-751], [V2G2-376], [V2G2-380], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_SessionSetup_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_001	

Table 404 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_001'.

Table **405 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_002'**

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-488], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_002	

Table 405 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **406 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_003'**

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-488], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_002	

Table 406 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_003'.

Table **407 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_004'**

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-488], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_002	

Table 407 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **408 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_005'**

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_005
Test objective	Test System executes GoodCase procedure, sends a valid SessionSetupRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a ServiceDiscoveryReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-488], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_003	

Table 408 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_005'.

Table **409 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_006'**

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_006
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', a valid PaymentOption and SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_004	

Table 409 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **410 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_007'**

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_007
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', a valid PaymentOption and SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_004	

Table 410 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_007'.

Table **411 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_008'**

TC Id	TC_EVCC_CMN_VTB_ServiceDiscovery_008
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of ServiceDiscoveryRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDiscovery_005	

Table 411 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDiscovery_008'.

10.3.3.5 EVCC test cases for ServiceDetail

10.3.3.5.1 General information

The following sub clauses include all *V2G ServiceDetail message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.5.2 Common test cases

Table **412 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_001'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_001
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.8.4.2.1; 8.4.3.4.1; 8.4.2; 8.6.2.3.2; 8.6.2.4.2; 8.6.3.5; 8.6.3.6; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-422], [V2G2-489], [V2G2-197], [V2G2-198], [V2G2-747], [V2G2-384], [V2G2-396], [V2G2-415], [V2G2-417], [V2G2-424], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_001	

Table 412 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_001'.

Table **413 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_002'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_002
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.8.4.2.1; 8.4.3.4.1; 8.4.2; 8.6.2.3.1; 8.6.2.4.1; 8.6.3.5; 8.6.3.6; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-422], [V2G2-489], [V2G2-197], [V2G2-198], [V2G2-747], [V2G2-378], [V2G2-392], [V2G2-415], [V2G2-417], [V2G2-424], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_002	

Table 413 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **414 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_003'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 414 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_003'.

Table **415 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_004'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 415 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **416 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_005'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 416 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_005'.

Table **417 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_006'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 417 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **418 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_007'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 418 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_007'.

Table **419 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_008'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 419 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **420 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_009'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_009
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_ServiceIDInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 420 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_009'.

Table **421 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_010'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_010
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ServiceDetailRes message with the current SessionID, ResponseCode 'FAILED_ServiceIDInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_003	

Table 421 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **422 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_011'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_011
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_004	

Table 422 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_011'.

Table **423 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_012'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_012
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a ServiceDetailReq message with the current SessionID, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-491], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_004	

Table 423 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **424 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_013'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_013
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of ServiceDetailRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_005	

Table 424 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_013'.

Table **425 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_014'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_014
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of ServiceDetailRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_005	

Table 425 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **426 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_015'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_015
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_006	

Table 426 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_015'.

Table **427 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_016'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_016
Test objective	Test System executes GoodCase procedure, signals CP State E and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_006	

Table 427 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **428 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_017'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_017
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_006	

Table 428 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_017'.

Table **429 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_018'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetail_018
Test objective	Test System executes GoodCase procedure, signals CP State F and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetail_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetail_006	

Table 429 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetail_018'.

10.3.3.6 EVCC test cases for ServiceDetailAndPaymentSelection

10.3.3.6.1 General information

The following sub clauses include all *V2G ServiceDetailAndPaymentSelection message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

10.3.3.6.2 Common test cases

Table **430 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'Contract', valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_001	

Table 430 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001'.

Table **431 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'ExternalPayment', valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.4.1; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_002	

Table 431 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **432 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 432 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003'.

Table **433 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 433 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **434 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 434 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005'.

Table **435 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_PaymentSelectionInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 435 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **436 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_ServiceSelectionInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 436 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007'.

Table **437 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'FAILED_NoChargeServiceSelected' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005	

Table 437 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **438 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009
Test objective	Test System executes GoodCase procedure and sends a valid ServiceDiscoveryRes or ServiceDetailRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving the PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-492], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_006	

Table 438 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009'.

Table **439 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010
Test objective	Test System executes GoodCase procedure and sends a valid AuthorizationRes message instead of PaymentServiceSelectionRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_007	

Table 439 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **440 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_008	

Table 440 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011'.

Table **441 - Test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012'**

TC Id	TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_008	

Table 441 lists the test case description for 'TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.6.3 AC specific test cases

Table **442 - Test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 442 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **443 - Test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012	

Table 443 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **444 - Test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 444 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **445 - Test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004'**

TC Id	`TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS == unknown, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012`	

Table 445 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **446 - Test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 446 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **447 - Test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012	

Table 447 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **448 - Test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007'**

TC Id	TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 448 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **449 - Test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008'**

TC Id	`TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS == unknown`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012`	

Table 449 lists the test case description for 'TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.6.4 DC specific test cases

Table **450 - Test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001'**

TC Id	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 450 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **451 - Test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002'**

TC Id	`TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS == unknown, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012`	

Table 451 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **452 - Test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003'**

TC Id	`TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011`	

Table 452 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **453 - Test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004'**

TC Id	`TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS == unknown, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012`	

Table 453 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **454 - Test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005'**

TC Id	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 454 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **455 - Test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006'**

TC Id	`TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS == unknown`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012`	

Table 455 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **456 - Test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007'**

TC Id	TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS == unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011	

Table 456 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **457 - Test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008'**

TC Id	`TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and ServiceList and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters. If a ServiceDetailReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS == unknown`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012`	

Table 457 lists the test case description for 'TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008'.

10.3.3.7 EVCC test cases for PaymentServiceSelection

10.3.3.7.1 General information

The following sub clauses include all *V2G PaymentServiceSelection message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.7.2 Common test cases

Table **458 - Test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_001'**

TC Id	TC_EVCC_CMN_VTB_PaymentServiceSelection_001
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'Contract', valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.2.4.2; 8.6.3.6; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-394], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_003	

Table 458 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_001'.

Table **459 - Test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_002'**

TC Id	TC_EVCC_CMN_VTB_PaymentServiceSelection_002
Test objective	Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'ExternalPayment', valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.4.1; 8.6.3.6; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-390], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_004	

Table 459 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **460 - Test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_003'**

TC Id	TC_EVCC_CMN_VTB_PaymentServiceSelection_003
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'Contract', valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_001	

Table 460 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_003'.

Table **461 - Test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_004'**

TC Id	TC_EVCC_CMN_VTB_PaymentServiceSelection_004
Test objective	Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'ExternalPayment', valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.4.1; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_002	

Table 461 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentServiceSelection_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.7.3 AC specific test cases

Table **462 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_001'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_001
Test objective	Test System executes GoodCase procedure, initiates an oscillator shutdown, sets a nominal duty cycle of 53% (32A) and sends a ServiceDiscoveryRes with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, SelectedPaymentOption 'ExternalPayment', valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.4.1; 8.6.2.1; 8.6.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-828]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_EVCC_CMN_PR_ServiceDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PaymentServiceSelection_001	

Table 462 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **463 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_002'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 463 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **464 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_003'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_003
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 464 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **465 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_004'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 465 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **466 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_005'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_005
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 466 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **467 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_006'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_006
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 467 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **468 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_007'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_007
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 468 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **469 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_008'**

TC Id	`TC_EVCC_AC_VTB_PaymentServiceSelection_008`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003`	

Table 469 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **470 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_009'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_009
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 470 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_009'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **471 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_010'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_010
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 471 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **472 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_011'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_011
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 472 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_011'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **473 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_012'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_012
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase.After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 473 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **474 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_013'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_013
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 474 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_013'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **475 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_014'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_014
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 475 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **476 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_015'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_015
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 476 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_015'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **477 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_016'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 477 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **478 - Test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_017'**

TC Id	TC_EVCC_AC_VTB_PaymentServiceSelection_017
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.3.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-376], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 478 lists the test case description for 'TC_EVCC_AC_VTB_PaymentServiceSelection_017'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.7.4 DC specific test cases

Table **479 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_001'**

TC Id	`TC_EVCC_DC_VTB_PaymentServiceSelection_001`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009`	

Table 479 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **480 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_002'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_002
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 480 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **481 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_003'**

TC Id	`TC_EVCC_DC_VTB_PaymentServiceSelection_003`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003`	

Table 481 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **482 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_004'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_004
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 482 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **483 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_005'**

TC Id	`TC_EVCC_DC_VTB_PaymentServiceSelection_005`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009`	

Table 483 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **484 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_006'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_006
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 484 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **485 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_007'**

TC Id	`TC_EVCC_DC_VTB_PaymentServiceSelection_007`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003`	

Table 485 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **486 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_008'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_008
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 486 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **487 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_009'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_009
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 487 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_009'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **488 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_010'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_010
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 488 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **489 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_011'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_011
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 489 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_011'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **490 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_012'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_012
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 490 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **491 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_013'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_013
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009	

Table 491 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_013'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **492 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_014'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_014
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDetailRes message with the current SessionID, ResponseCode 'OK', requested ServiceID and all additional mandatory parameters. If another ServiceDetailReq message is received, Test System then sends a new valid ServiceDetailRes message with the corresponding ServiceID. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.3.6; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-493], [V2G2-494], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-432], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := serviceDetail, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010	

Table 492 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **493 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_015'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_015
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'Contract', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.2; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003	

Table 493 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_015'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **494 - Test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_016'**

TC Id	TC_EVCC_DC_VTB_PaymentServiceSelection_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a ServiceDiscoveryRes message with the current SessionID, ResponseCode 'Ok', PaymentOption 'ExternalPayment', valid SupportedEnergyTransferMode and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct PaymentServiceSelectionReq message with the current SessionID, the previous SelectedPaymentOption, valid ServiceID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.4.1; 8.4.3.4.2; 8.4.3.5.2; 8.8.4.2.1; 8.4.3.5.3; 8.8.4.2.1; 8.5.2.24; 8.5.2.25; 8.6.2.4.1; 8.6.2.1; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-201], [V2G2-202], [V2G2-490], [V2G2-431], [V2G2-747], [V2G2-349], [V2G2-350], [V2G2-351], [V2G2-352], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662] [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_VAS := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_SessionSetupPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004	

Table 494 lists the test case description for 'TC_EVCC_DC_VTB_PaymentServiceSelection_016'.

10.3.3.8 EVCC test cases for CertificateInstallation

10.3.3.8.1 General information

The following sub clauses include all *V2G CertificateInstallation message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.8.2 Common test cases

Table **495 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_001'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_001
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct CertificateInstallationReq message with a valid signature, current SessionID, OEMProvisioning certificate and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.8.4.2.1; 8.4.2; 8.4.3.11.2; 8.5.2.27; 8.6.2.3.2; 8.6.2.4.2; 8.6.3.3; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.5.2.26; 7.3.3; 7.9.2.4.3; AnnexF
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-235], [V2G2-236], [V2G2-496], [V2G2-747], [V2G2-355], [V2G2-356], [V2G2-386], [V2G2-398], [V2G2-411], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-893], [V2G2-909], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-822], [V2G2-932], [V2G2-933]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_001	

Table 495 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_001'.

Table **496 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_002'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 496 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **497 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_003'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 497 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_003'.

Table **498 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_004'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 498 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **499 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_005'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 499 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_005'.

Table **500 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_006'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_CertificateExpired' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 500 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **501 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_007'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_CertificateRevoked' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 501 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_007'.

Table **502 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_008'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateInstallationRes message with current SessionID, ResponseCode 'FAILED_NoCertificateAvailable' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_002	

Table 502 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **503 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_009'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_009
Test objective	Test System executes GoodCase procedure, sends a valid PaymentServiceSelectionRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a CertificateInstallationReq message with a valid signature, current SessionID, OEMProvisioning certificate and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-498], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_003	

Table 503 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_009'.

Table **504 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_010'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of CertificateInstallationRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_004	

Table 504 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **505 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_011'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_011
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a CertificateInstallationRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_005	

Table 505 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_011'.

Table **506 - Test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_012'**

TC Id	TC_EVCC_CMN_VTB_CertificateInstallation_012
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a CertificateInstallationRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateInstallation_005	

Table 506 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateInstallation_012'.

10.3.3.9 EVCC test cases for CertificateUpdate

10.3.3.9.1 General information

The following sub clauses include all *V2G CertificateUpdate message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.9.2 Common test cases

Table **507 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_001'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_001
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct CertificateUpdateReq message with a valid signature, current SessionID, contract certificate chain, valid EMAID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.8.4.2.1; 8.4.2; 8.4.3.10.1; 8.4.3.10.2; 8.5.2.5; 8.5.2.27; 8.6.2.3.2; 8.6.2.4.2; 8.6.3.4; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.5.2.26; 7.3.3; AnnexF; 7.9.2.4.3; 7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-227], [V2G2-228], [V2G2-229], [V2G2-274], [V2G2-275], [V2G2-497], [V2G2-747], [V2G2-355], [V2G2-356], [V2G2-388], [V2G2-400], [V2G2-413], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-888], [V2G2-909], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-884], [V2G2-822], [V2G2-103], [V2G2-104]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_001	

Table 507 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_001'.

Table **508 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_002'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 508 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **509 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_003'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 509 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_003'.

Table **510 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_004'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 510 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **511 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_005'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 511 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_005'.

Table **512 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_006'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_CertificateExpired' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 512 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **513 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_007'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_CertificateRevoked' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 513 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_007'.

Table **514 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_008'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_NoCertificateAvailable' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 514 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **515 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_009'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_009
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_ContractCanceled' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 515 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_009'.

Table **516 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_010'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_010
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CertificateUpdateRes message with current SessionID, ResponseCode 'FAILED_ChallengeInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_002	

Table 516 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **517 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_011'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_011
Test objective	Test System executes GoodCase procedure, sends a valid PaymentServiceSelectionRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a CertificateUpdateReq message with a valid signature, current SessionID, valid contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-499], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_003	

Table 517 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_011'.

Table **518 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_012'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_012
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of CertificateUpdateRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_004	

Table 518 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **519 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_013'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_013
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a CertificateUpdateRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_005	

Table 519 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_013'.

Table **520 - Test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_014'**

TC Id	TC_EVCC_CMN_VTB_CertificateUpdate_014
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a CertificateUpdateRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateUpdate_005	

Table 520 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateUpdate_014'.

10.3.3.10 EVCC test cases for CertificateHandlingAndPaymentDetails

10.3.3.10.1 General information

The following sub clauses include all *V2G CertificateHandlingAndPaymentDetails message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.10.2 Common test cases

Table **521 - Test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionRes with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. If a valid CertificateInstallationReq or CertificateUpdateReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.6.2.3.2; 8.6.2.4.2; 8.6.2.1; 8.5.2.26; 7.3.2; 7.3.3; AnnexF; 7.9.2.4.3; 7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-205], [V2G2-206], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-898], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926]; [V2G2-884], [V2G2-822], [V2G2-103], [V2G2-104]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_001	

Table 521 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001'.

Table **522 - Test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-502], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002	

Table 522 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **523 - Test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-502], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002	

Table 523 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003'.

Table **524 - Test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PaymentDetailsRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-502], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002	

Table 524 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **525 - Test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes, CertificateInstallationRes or CertificateUpdateRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-502], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_003	

Table 525 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005'.

Table **526 - Test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PaymentDetailsRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_004	

Table 526 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **527 - Test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_005	

Table 527 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007'.

Table **528 - Test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008'**

TC Id	TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a PaymentDetailsRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_005	

Table 528 lists the test case description for 'TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008'.

10.3.3.11 EVCC test cases for PaymentDetails

10.3.3.11.1 General information

The following sub clauses include all *V2G PaymentDetails message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

10.3.3.11.2 Common test cases

Table **529 - Test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_001'**

TC Id	TC_EVCC_CMN_VTB_PaymentDetails_001
Test objective	Test System executes GoodCase procedure and sends a CertificateInstallationRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.8.4.2; 8.4.2; 8.6.2.3.2; 8.6.2.4.2; 8.6.2.1; 7.3.2; 7.9.2.3; 7.9.2.4.5; 8.5.2.26; 7.3.3; AnnexF; 8.4.3.11.3; 7.9.2.4.3; 7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-205], [V2G2-206], [V2G2-500], [V2G2-747], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-108], [V2G2-652], [V2G2-653], [V2G2-898], [V2G2-909], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-884], [V2G2-896], [V2G2-822], [V2G2-103], [V2G2-104]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateInstallation_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentDetails_001	

Table 529 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_001'.

Table **530 - Test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_002'**

TC Id	TC_EVCC_CMN_VTB_PaymentDetails_002
Test objective	Test System executes GoodCase procedure and sends a CertificateUpdateRes message with a valid signature, current SessionID, ResponseCode 'OK', SAProvisioning certificate chain, contract certificate chain, ContractSignatureEncryptedPrivateKey, DHpublickey, EMAID and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.8.4.2; 8.4.2; 8.6.2.3.2; 8.6.2.4.2; 8.6.2.1; 7.3.2; 7.9.2.3; 7.9.2.4.5; 8.5.2.26; 7.3.3; 8.4.3.10.3; 7.9.2.4.3; 7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-205], [V2G2-206], [V2G2-501], [V2G2-747], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-108], [V2G2-652], [V2G2-653], [V2G2-898], [V2G2-909], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-891], [V2G2-822], [V2G2-103], [V2G2-104]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateUpdate_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentDetails_002	

Table 530 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **531 - Test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_003'**

TC Id	TC_EVCC_CMN_VTB_PaymentDetails_003
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct PaymentDetailsReq message with the current SessionID, valid EMAID, contract certificate chain and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.6.2; 8.8.4.2; 8.4.2; 8.6.2.3.2; 8.6.2.4.2; 8.6.2.1; 8.5.2.26, 7.3.2; 7.3.3; 7.9.2.4.3; 7.9.2.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-205], [V2G2-206], [V2G2-495], [V2G2-747], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-898], [V2G2-353], [V2G2-354], [V2G2-004], [V2G2-009], [V2G2-656], [V2G2-926], [V2G2-822], [V2G2-103], [V2G2-104]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := none_, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_PaymentDetails_003	

Table 531 lists the test case description for 'TC_EVCC_CMN_VTB_PaymentDetails_003'.

10.3.3.12 EVCC test cases for Authorization

10.3.3.12.1 General information

The following sub clauses include all *V2G Authorization message* test case descriptions, where the SUT is the EVCC.

10.3.3.12.2 Common test cases

Table **532 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_001'**

TC Id	TC_EVCC_CMN_VTB_Authorization_001
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends a correct AuthorizationReq message with a valid signature, the current SessionID, valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.1; 8.7.4.2; 8.8.4.2.1; 8.8.4.2.2; 8.6.2.3.2; 8.6.2.4.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-210], [V2G2-211], [V2G2-747], [V2G2-503], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-900], [V2G2-909], [V2G2-684]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_001	

Table 532 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **533 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_002'**

TC Id	TC_EVCC_CMN_VTB_Authorization_002
Test objective	Test System executes GoodCase procedure and sends a PaymentServiceSelectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct AuthorizationReq message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.1; 8.7.4.2; 8.8.4.2.1; 8.8.4.2.2; 8.6.2.3.1; 8.6.2.4.1; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-210], [V2G2-211], [V2G2-747], [V2G2-509], [V2G2-376], [V2G2-390], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_001	

Table 533 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_002'.

Table **534 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_003'**

TC Id	TC_EVCC_CMN_VTB_Authorization_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_002	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_003	

Table 534 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **535 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_004'**

TC Id	TC_EVCC_CMN_VTB_Authorization_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_002	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_003	

Table 535 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_004'.

Table **536 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_005'**

TC Id	TC_EVCC_CMN_VTB_Authorization_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_002	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_003	

Table 536 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **537 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_006'**

TC Id	TC_EVCC_CMN_VTB_Authorization_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_ChallengeInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_002	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_003	

Table 537 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_006'.

Table **538 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_007'**

TC Id	TC_EVCC_CMN_VTB_Authorization_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending an AuthorizationRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_Authorization_002	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_003	

Table 538 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **539 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_008'**

TC Id	TC_EVCC_CMN_VTB_Authorization_008
Test objective	Test System executes GoodCase procedure and sends a valid ServicePaymentSelectionRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a AuthorizationReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-504], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_002	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_004	

Table 539 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_008'.

Table **540 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_009'**

TC Id	TC_EVCC_CMN_VTB_Authorization_009
Test objective	Test System executes GoodCase procedure, sends continously an AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing' and all additional mandatory parameters and waits for another AuthorizationReq message with the current SessionID and all additional mandatory parameters until the V2G_EVCC_Ongoing_Timer is equal or larger than V2G_EVCC_Ongoing_Timeout. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.3.1; 8.7.3.4; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-605], [V2G2-710], [V2G2-711], [V2G2-684], [V2G2-845]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_002	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_005	

Table 540 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_009'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **541 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_010'**

TC Id	TC_EVCC_CMN_VTB_Authorization_010
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends a correct AuthorizationReq message with a valid signature, the current SessionID, valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.1; 8.7.4.2; 8.8.4.2.1; 8.8.4.2.2; 8.6.2.3.2; 8.6.2.4.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 7.9.2.4.3; 8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-210], [V2G2-211], [V2G2-747], [V2G2-503], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-900], [V2G2-909], [V2G2-121], [V2G2-122], [V2G2-817], [V2G2-818], [V2G2-896]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_001	

Table 541 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_010'.

Table **542 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_011'**

TC Id	TC_EVCC_CMN_VTB_Authorization_011
Test objective	Test System executes GoodCase procedure and sends a PaymentDetailRes message with the current SessionID, ResponseCode 'OK', valid GenChallenge and all additional mandatory parameters. Test System then checks that the SUT sends a correct AuthorizationReq message with a valid signature, the current SessionID, valid GenChallenge and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.7.1; 8.7.4.2; 8.8.4.2.1; 8.8.4.2.2; 8.6.2.3.2; 8.6.2.4.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 7.9.2.4.3; 8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-210], [V2G2-211], [V2G2-747], [V2G2-503], [V2G2-380], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-900], [V2G2-909], [V2G2-121], [V2G2-122], [V2G2-230], [V2G2-817], [V2G2-818], [V2G2-827], [V2G2-891]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC
PIXIT selection	PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_001	

Table 542 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_011'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **543 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_012'**

TC Id	TC_EVCC_CMN_VTB_Authorization_012
Test objective	Test System executes GoodCase procedure and sends a vaild PaymentServiceSelectionRes message instead of AuthorizationRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_002	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_006	

Table 543 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_012'.

Table **544 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_013'**

TC Id	TC_EVCC_CMN_VTB_Authorization_013
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_002	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_007	

Table 544 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_013'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **545 - Test case description for 'TC_EVCC_CMN_VTB_Authorization_014'**

TC Id	TC_EVCC_CMN_VTB_Authorization_014
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_Authorization_002	
Expected behavior	
f_EVCC_CMN_TB_VTB_Authorization_007	

Table 545 lists the test case description for 'TC_EVCC_CMN_VTB_Authorization_014'.

10.3.3.13 EVCC test cases for ChargeParameterDiscovery

10.3.3.13.1 General information

The following sub clauses include all *V2G ChargeParameterDiscovery message* test case descriptions, where the SUT is the EVCC.

10.3.3.13.2 Common test cases

Table **546 - Test case description for 'TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001'**

TC Id	TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a valid AuthorizationRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_002	

Table 546 lists the test case description for 'TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **547 - Test case description for 'TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002'**

TC Id	TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of ChargeParameterDiscoveryRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_003	

Table 547 lists the test case description for 'TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002'.

10.3.3.13.3 AC specific test cases

Table **548 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_001'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := unknown
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_001	

Table 548 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **549 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_002'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 549 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_002'.

Table **550 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_003'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 550 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **551 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_004'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 551 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_004'.

Table **552 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_005'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongEnergyTransferType' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 552 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **553 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_006'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongChargeParameter' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001	

Table 553 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_006'.

Table **554 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_007'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_007
Test objective	Test System executes GoodCase procedure, sends continously a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing', and all additional mandatory parameters and waits for another valid ChargeParameterDiscoveryReq message until the V2G_EVCC_Ongoing_Timer is equal or larger than V2G_EVCC_Ongoing_Timeout. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.3.1; 8.7.3.4
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-605], [V2G2-710], [V2G2-711], [V2G2-685]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_002	

Table 554 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **555 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_008'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_008
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_CMN_DepartureTime := unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_003	

Table 555 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_008'.

Table **556 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_009'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_009
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_004	

Table 556 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_009'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **557 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_010'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_010
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_004	

Table 557 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_010'.

Table **558 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_011'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_011
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_CMN_DepartureTime := unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_003	

Table 558 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_011'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **559 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_012'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_012
Test objective	Test System executes GoodCase procedure and sends a AuthorizationRes message with with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := none_
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_005	

Table 559 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_012'.

Table **560 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_013'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_013
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_CMN_DepartureTime := none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 560 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_013'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **561 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_014'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_014
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_CMN_DepartureTime := none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_006	

Table 561 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_014'.

Table **562 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_015'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_015
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 562 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_015'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **563 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_016'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 563 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **564 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_017'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_017
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating an oscillator B1/B2 transition and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := departureTime, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 564 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_017'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **565 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_018'**

TC Id	`TC_EVCC_AC_VTB_ChargeParameterDiscovery_018`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := aC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_DepartureTime := unknown, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004`	

Table 565 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_018'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **566 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_019'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_019
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 566 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_019'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **567 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_020'**

TC Id	`TC_EVCC_AC_VTB_ChargeParameterDiscovery_020`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := aC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_DepartureTime := departureTime, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004`	

Table 567 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_020'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **568 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_021'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_021
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := unknown
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 568 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_021'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **569 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_022'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_022
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 569 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_022'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **570 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_023'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_023
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := departureTime
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 570 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_023'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **571 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_024'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_024
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := unknown
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 571 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_024'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **572 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_025'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_025
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 572 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_025'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **573 - Test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_026'**

TC Id	TC_EVCC_AC_VTB_ChargeParameterDiscovery_026
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.2.7; 8.5.3.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.2.1; 8.7.4.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-360], [V2G2-361], [V2G2-376], [V2G2-380], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-930], [V2G2-832], [V2G2-215], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := departureTime
PreCondition	
f_checkWakeUpConfig, f_EVCC_AC_PR_SessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 573 lists the test case description for 'TC_EVCC_AC_VTB_ChargeParameterDiscovery_026'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.13.4 DC specific test cases

Table **574 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_001'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_001
Test objective	Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := unknown
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_001	

Table 574 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_001'.

Table **575 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_002'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 575 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **576 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_003'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 576 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_003'.

Table **577 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_004'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 577 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **578 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_005'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongEnergyTransferType' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 578 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_005'.

Table **579 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_006'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'FAILED_WrongChargeParameter' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-506], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001	

Table 579 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **580 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_007'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_007
Test objective	Test System executes GoodCase procedure, sends continously a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing', and all additional mandatory parameters and waits for another valid ChargeParameterDiscoveryReq message until the V2G_EVCC_Ongoing_Timer is equal or larger than V2G_EVCC_Ongoing_Timeout. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.3.1; 8.7.3.4
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-605], [V2G2-710], [V2G2-711], [V2G2-685]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_002	

Table 580 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_007'.

Table **581 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_008'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_008
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_CMN_DepartureTime := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_003	

Table 581 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **582 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_009'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_009
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_004	

Table 582 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_009'.

Table **583 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_010'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_010
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_004	

Table 583 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **584 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_011'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_011
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_CMN_DepartureTime := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_003	

Table 584 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_011'.

Table **585 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_012'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_012
Test objective	Test System executes GoodCase procedure and sends a AuthorizationRes message with with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := none_
PreCondition	
f_EVCC_CMN_PR_Authorization_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_005	

Table 585 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **586 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_013'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_013
Test objective	Test System executes GoodCase procedure, initiates a renegotiation and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_CMN_DepartureTime := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 586 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_013'.

Table **587 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_014'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_014
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters if a PowerDeliveryReq message with the current SessionID, ChargeProgress 'renegotiate' and all additional mandatory parameters was received before. Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, a valid RequestedEnergyTransferType, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-683], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-215]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_CMN_DepartureTime := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_006	

Table 587 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **588 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_015'**

TC Id	`TC_EVCC_DC_VTB_ChargeParameterDiscovery_015`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_DepartureTime := unknown, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004`	

Table 588 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_015'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **589 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_016'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_016
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 589 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **590 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_017'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_017
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PICS_CMN_CMN_WakeUp' Test System resumes the previously paused session by initiating 5% duty cycle and waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' should be in the range of [10.0s - 60.0s] to avoid any SUT-initiated wake-up Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := departureTime, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 590 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_017'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **591 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_018'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_018
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 591 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_018'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **592 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_019'**

TC Id	`TC_EVCC_DC_VTB_ChargeParameterDiscovery_019`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_DepartureTime := none_, PIXIT_EVCC_CMN_Pause := pause`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005`	

Table 592 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_019'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **593 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_020'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_020
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'PIXIT_CMN_CMN_WakeUp' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time for the SUT-initiated wake-up (PIXIT_CMN_CMN_WakeUp) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := departureTime, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 593 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_020'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **594 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_021'**

TC Id	`TC_EVCC_DC_VTB_ChargeParameterDiscovery_021`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_DepartureTime := unknown`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004`	

Table 594 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_021'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **595 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_022'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_022
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 595 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_022'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **596 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_023'**

TC Id	`TC_EVCC_DC_VTB_ChargeParameterDiscovery_023`
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	`PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true`
PIXIT selection	`PIXIT_EVCC_CMN_DepartureTime := departureTime`
PreCondition	
`f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001`	
Expected behavior	
`f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004`	

Table 596 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_023'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **597 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_024'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_024
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := unknown
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 597 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_024'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **598 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_025'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_025
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, without a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := none_
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005	

Table 598 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_025'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **599 - Test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_026'**

TC Id	TC_EVCC_DC_VTB_ChargeParameterDiscovery_026
Test objective	Test System executes GoodCase procedure and waits for a paused V2G communication session by receiving a SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). After sending of a valid SessionStopRes message, Test System turns off the CP oscillator to reach X1 state and checks whether the SUT signals CP State B within the entire sleeping phase. After 'par_SECC_Pmax0W' the SUT resumes the previously paused session by initiating a BCB toggle. Furthermore Test System waits for successful link detection (old logical network parameter) if the data link was terminated before (Unpowering the low-layer communication module is optional). Test System executes GoodCase procedure and sends a AuthorizationRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. The value for 'PICS_CMN_CMN_WakeUp' shall be set to a higher value than the time interval for a Pmax element equals to 0W within SAScheduleList (par_SECC_Pmax0W) Test System then checks that the SUT sends a correct ChargeParameterDiscoveryReq message with the current SessionID, the previous RequestedEnergyTransferMode, a DepartureTime and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.8.2; 8.8.4.2.1; 8.5.4.3 8.5.2.7; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-214], [V2G2-216], [V2G2-505], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-370], [V2G2-371], [V2G2-390], [V2G2-394], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-740], [V2G2-744], [V2G2-742]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge, par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp, PICS_CMN_CMN_CombinedTesting := true
PIXIT selection	PIXIT_EVCC_CMN_DepartureTime := departureTime
PreCondition	
f_checkWakeUpConfig, f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002, f_EVCC_startSleepingPhase, f_EVCC_CMN_PR_AuthorizationPausedSession_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004	

Table 599 lists the test case description for 'TC_EVCC_DC_VTB_ChargeParameterDiscovery_026'.

10.3.3.14 EVCC test cases for PowerDelivery

10.3.3.14.1 General information

The following sub clauses include all *V2G PowerDelivery message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.14.2 AC specific test cases

Table **600 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_001'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_001
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-673], [V2G2-846]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_001	

Table 600 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_001'.

Table **601 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_002'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 601 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **602 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_003'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 602 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_003'.

Table **603 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_004'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 603 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **604 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_005'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ChargingProfileInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 604 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_005'.

Table **605 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_006'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_TariffSelectionInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 605 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **606 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_007'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_PowerDeliveryNotApplied' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 606 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_007'.

Table **607 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_008'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ContactorError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_002	

Table 607 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **608 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_009'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_009
Test objective	Test System executes GoodCase procedure and sends a valid ChargeParameterDiscoveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_003	

Table 608 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_009'.

Table **609 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_010'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PowerDeliveryRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_004	

Table 609 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **610 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_011'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_011
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_005	

Table 610 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_011'.

Table **611 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_012'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_012
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_005	

Table 611 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **612 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_013'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_013
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 612 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_013'.

Table **613 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_014'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_014
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 613 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **614 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_015'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_015
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-829], [V2G2-293], [V2G2-673], [V2G2-846]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_001	

Table 614 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_015'.

Table **615 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_016'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_016
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 615 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **616 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_017'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_017
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 616 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_017'.

Table **617 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_018'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_018
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-296], [V2G2-675], [V2G2-673], [V2G2-846]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_001	

Table 617 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_018'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **618 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_019'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_019
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including MaxNumberOfPhasesInUse and the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-829], [V2G2-293], [V2G2-296], [V2G2-675], [V2G2-673], [V2G2-846]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_001	

Table 618 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_019'.

Table **619 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_020'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_020
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default SAScheduleTuple and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-675], [V2G2-673], [V2G2-846]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := none_
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_001	

Table 619 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_020'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **620 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_021'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_021
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and the default SAScheduleTuple and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State C or D within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-846], [V2G2-510], [V2G2-847], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-829], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-675], [V2G2-673], [V2G2-846]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization := none_
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_001	

Table 620 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_021'.

Table **621 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_022'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_022
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffsand all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-296], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 621 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_022'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **622 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_023'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_023
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-296], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := false, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 622 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_023'.

Table **623 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_024'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_024
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including MaxNumberOfPhasesInUse and the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-296], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 623 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_024'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **624 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_025'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_025
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including MaxNumberOfPhasesInUse and the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-296], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := false, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 624 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_025'.

Table **625 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_026'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_026
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffsand all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := none_, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 625 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_026'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **626 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_027'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_027
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := false, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := none , PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 626 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_027'.

Table **627 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_028'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_028
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization := none , PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 627 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_028'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **628 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_029'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_029
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList including three SAScheduleTuples with three different price levels and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including MaxNumberOfPhasesInUse and the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := false, PICS_EVCC_CMN_SalesTariff := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization := none , PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 628 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_029'.

Table **629 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_030'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_030
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := oldSAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 629 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_030'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **630 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_031'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_031
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := oldSAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 630 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_031'.

Table **631 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_032'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_032
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including MaxNumberOfPhasesInUse and the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := oldSAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 631 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_032'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **632 - Test case description for 'TC_EVCC_AC_VTB_PowerDelivery_033'**

TC Id	TC_EVCC_AC_VTB_PowerDelivery_033
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including MaxNumberOfPhasesInUse and the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2; 8.4.3.9.3; 8.8.4.2.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1; 8.7.4.3; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-836], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286] [V2G2-510], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-829], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := maxNumberOfPhasesInUse, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := oldSAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001	
Expected behavior	
f_EVCC_AC_TB_VTB_PowerDelivery_006	

Table 632 lists the test case description for 'TC_EVCC_AC_VTB_PowerDelivery_033'.

10.3.3.15 EVCC test cases for SessionStop

10.3.3.15.1 General information

The following sub clauses include all *V2G SessionStop message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.15.2 AC specific test cases

Table **633 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_001'**

TC Id	TC_EVCC_AC_VTB_SessionStop_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.8.4.2.1; 8.6.2.1
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-508], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 633 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_001'.

Table **634 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_002'**

TC Id	TC_EVCC_AC_VTB_SessionStop_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_003	

Table 634 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **635 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_003'**

TC Id	TC_EVCC_AC_VTB_SessionStop_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_003	

Table 635 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_003'.

Table **636 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_004'**

TC Id	TC_EVCC_AC_VTB_SessionStop_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_003	

Table 636 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **637 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_005'**

TC Id	TC_EVCC_AC_VTB_SessionStop_005
Test objective	Test System executes GoodCase procedure, sends a valid PowerDeliveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a SessionStopReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_004	

Table 637 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_005'.

Table **638 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_006'**

TC Id	TC_EVCC_AC_VTB_SessionStop_006
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of SessionStopRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_SessionStop_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_005	

Table 638 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **639 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_007'**

TC Id	TC_EVCC_AC_VTB_SessionStop_007
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.4.2.2; 8.6.2.3.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.3.4
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-633], [V2G2-635], [V2G2-637]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 639 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_007'.

Table **640 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_008'**

TC Id	TC_EVCC_AC_VTB_SessionStop_008
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.4.2.2; 8.6.2.3.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 640 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **641 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_009'**

TC Id	TC_EVCC_AC_VTB_SessionStop_009
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.4
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 641 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_009'.

Table **642 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_010'**

TC Id	TC_EVCC_AC_VTB_SessionStop_010
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.4.2.2; 8.6.2.3.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.3.4; 7.4
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 642 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **643 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_011'**

TC Id	TC_EVCC_AC_VTB_SessionStop_011
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.4.2.2; 8.6.2.3.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.4
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 643 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_011'.

Table **644 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_012'**

TC Id	TC_EVCC_AC_VTB_SessionStop_012
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 644 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **645 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_013'**

TC Id	TC_EVCC_AC_VTB_SessionStop_013
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.4.2.2; 8.6.2.3.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 645 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_013'.

Table **646 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_014'**

TC Id	TC_EVCC_AC_VTB_SessionStop_014
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.4.2.2; 8.6.2.3.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 646 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **647 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_015'**

TC Id	TC_EVCC_AC_VTB_SessionStop_015
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 647 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_015'.

Table **648 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_016'**

TC Id	TC_EVCC_AC_VTB_SessionStop_016
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.4.2.2; 8.6.2.3.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 648 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **649 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_017'**

TC Id	TC_EVCC_AC_VTB_SessionStop_017
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.4.2.2; 8.6.2.3.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 649 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_017'.

Table **650 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_018'**

TC Id	TC_EVCC_AC_VTB_SessionStop_018
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge
PIXIT selection	PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 650 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_018'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **651 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_019'**

TC Id	TC_EVCC_AC_VTB_SessionStop_019
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.4.2.2; 8.6.2.3.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge
PIXIT selection	PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 651 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_019'.

Table **652 - Test case description for 'TC_EVCC_AC_VTB_SessionStop_020'**

TC Id	TC_EVCC_AC_VTB_SessionStop_020
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.4.2.2; 8.6.2.3.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-239], [V2G2-738], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-520], [V2G2-376], [V2G2-402], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge
PIXIT selection	PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_SessionStop_001	

Table 652 lists the test case description for 'TC_EVCC_AC_VTB_SessionStop_020'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.15.3 DC specific test cases

Table **653 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_001'**

TC Id	TC_EVCC_DC_VTB_SessionStop_001
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 653 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_001'.

Table **654 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_002'**

TC Id	TC_EVCC_DC_VTB_SessionStop_002
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 654 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **655 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_003'**

TC Id	TC_EVCC_DC_VTB_SessionStop_003
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 7.3.4; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 655 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_003'.

Table **656 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_004'**

TC Id	TC_EVCC_DC_VTB_SessionStop_004
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 7.3.4; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 656 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **657 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_005'**

TC Id	TC_EVCC_DC_VTB_SessionStop_005
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 657 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_005'.

Table **658 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_006'**

TC Id	TC_EVCC_DC_VTB_SessionStop_006
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 658 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **659 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_007'**

TC Id	TC_EVCC_DC_VTB_SessionStop_007
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 659 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_007'.

Table **660 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_008'**

TC Id	TC_EVCC_DC_VTB_SessionStop_008
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 660 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **661 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_009'**

TC Id	TC_EVCC_DC_VTB_SessionStop_009
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 7.3.4; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 661 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_009'.

Table **662 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_010'**

TC Id	TC_EVCC_DC_VTB_SessionStop_010
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 7.3.4; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 662 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **663 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_011'**

TC Id	TC_EVCC_DC_VTB_SessionStop_011
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 663 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_011'.

Table **664 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_012'**

TC Id	TC_EVCC_DC_VTB_SessionStop_012
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 664 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **665 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_013'**

TC Id	TC_EVCC_DC_VTB_SessionStop_013
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 665 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_013'.

Table **666 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_014'**

TC Id	TC_EVCC_DC_VTB_SessionStop_014
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 666 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **667 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_015'**

TC Id	TC_EVCC_DC_VTB_SessionStop_015
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 7.3.4; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 667 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_015'.

Table **668 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_016'**

TC Id	TC_EVCC_DC_VTB_SessionStop_016
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 7.3.4; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 668 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **669 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_017'**

TC Id	TC_EVCC_DC_VTB_SessionStop_017
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 669 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_017'.

Table **670 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_018'**

TC Id	TC_EVCC_DC_VTB_SessionStop_018
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 670 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_018'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **671 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_019'**

TC Id	TC_EVCC_DC_VTB_SessionStop_019
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 671 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_019'.

Table **672 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_020'**

TC Id	TC_EVCC_DC_VTB_SessionStop_020
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 672 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_020'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **673 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_021'**

TC Id	TC_EVCC_DC_VTB_SessionStop_021
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 7.3.4; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 673 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_021'.

Table **674 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_022'**

TC Id	TC_EVCC_DC_VTB_SessionStop_022
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 7.3.4; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 674 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_022'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **675 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_023'**

TC Id	TC_EVCC_DC_VTB_SessionStop_023
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 675 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_023'.

Table **676 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_024'**

TC Id	TC_EVCC_DC_VTB_SessionStop_024
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 676 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_024'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **677 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_025'**

TC Id	TC_EVCC_DC_VTB_SessionStop_025
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 677 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_025'.

Table **678 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_026'**

TC Id	TC_EVCC_DC_VTB_SessionStop_026
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 678 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_026'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **679 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_027'**

TC Id	TC_EVCC_DC_VTB_SessionStop_027
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 7.3.4; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 679 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_027'.

Table **680 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_028'**

TC Id	TC_EVCC_DC_VTB_SessionStop_028
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 7.3.4; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 680 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_028'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **681 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_029'**

TC Id	TC_EVCC_DC_VTB_SessionStop_029
Test objective	Test System executes GoodCase procedure and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-508], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection, PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002	

Table 681 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_029'.

Table **682 - Test case description for 'TC_EVCC_DC_VTB_SessionStop_030'**

TC Id	TC_EVCC_DC_VTB_SessionStop_030
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.12; 8.4.2; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.8.4.2.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-619], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := none_, PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_SessionStop_001	

Table 682 lists the test case description for 'TC_EVCC_DC_VTB_SessionStop_030'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.16 EVCC test cases for ChargingStatus

10.3.3.16.1 General information

The following sub clause includes all *V2G ChargingStatus message* test case descriptions, where the SUT is the EVCC.

10.3.3.16.2 AC specific test cases

Table **683 - Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_001'**

TC Id	TC_EVCC_AC_VTB_ChargingStatus_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT sends a correct ChargingStatusReq message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.4.2.2; 8.4.4.2.3; 8.8.4.2; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-242], [V2G2-514], [V2G2-747], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_PowerDeliveryStart_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_001	

Table 683 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_001'.

Table **684 - Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_002'**

TC Id	TC_EVCC_AC_VTB_ChargingStatus_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargingStatusRes message with the current sessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-511], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_002	

Table 684 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **685 - Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_003'**

TC Id	TC_EVCC_AC_VTB_ChargingStatus_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargingStatusRes message with the current sessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-511], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_002	

Table 685 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_003'.

Table **686 - Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_004'**

TC Id	TC_EVCC_AC_VTB_ChargingStatus_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a ChargingStatusRes message with the current sessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-511], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_002	

Table 686 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **687 - Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_005'**

TC Id	TC_EVCC_AC_VTB_ChargingStatus_005
Test objective	Test System executes GoodCase procedure and sends a valid PowerDeliveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a ChargingStatusReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-511], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_003	

Table 687 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_005'.

Table **688 - Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_006'**

TC Id	TC_EVCC_AC_VTB_ChargingStatus_006
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_004	

Table 688 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **689 - Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_007'**

TC Id	TC_EVCC_AC_VTB_ChargingStatus_007
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_004	

Table 689 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_007'.

Table **690 - Test case description for 'TC_EVCC_AC_VTB_ChargingStatus_008'**

TC Id	TC_EVCC_AC_VTB_ChargingStatus_008
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of ChargingStatusRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingStatus_005	

Table 690 lists the test case description for 'TC_EVCC_AC_VTB_ChargingStatus_008'.

10.3.3.17 EVCC test cases for ChargingLoopAcOrPowerDelivery

10.3.3.17.1 General information

The following sub clause includes all *V2G ChargingLoopAcOrPowerDelivery message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.17.2 AC specific test cases

Table **691 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. This process will be continued for 'PICS_CMN_CMN_LoopCounter' message sequences if a new valid ChargingStatusReq message was received afterwards. In the last sequence EVSENotification will be changed to 'StopCharging'. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State B within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.4.2.2; 8.4.4.2.3; 8.8.4.1; 8.8.4.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-513], [V2G2-848], [V2G2-242], [V2G2-516], [V2G2-679], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-849], [V2G2-832], [V2G2-293], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_ReceiptRequired := false
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_001	

Table 691 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001'.

Table **692 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Furthermore Test System sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'Ok' and all additional mandatory parameters if a valid MeteringReceiptReq message was received before. This process will be continued for 'PICS_CMN_CMN_LoopCounter' message sequences if a new valid ChargingStatusReq message was received afterwards. In the last charging loop sequence EVSENotification will be changed to 'StopCharging'. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters after receiving a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Furthermore Test System checks that the SUT applies CP State B within 'V2G_SECC_CPState_Detection_Timeout' afterwards.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.4.2.2; 8.4.4.2.3, 8.8.4.1; 8.8.4.2; 8.4.3.9.1; 8.6.2.3.2; 8.6.3.1; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-848], [V2G2-242], [V2G2-519], [V2G2-679], [V2G2-380], [V2G2-404], [V2G2-518], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-849], [V2G2-832], [V2G2-903], [V2G2-293], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_002	

Table 692 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **693 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003	

Table 693 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003'.

Table **694 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003	

Table 694 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **695 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003	

Table 695 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005'.

Table **696 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ContactorError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003	

Table 696 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **697 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007
Test objective	Test System executes GoodCase procedure, sends a valid ChargingStatusRes or MeteringReceiptRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_004	

Table 697 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007'.

Table **698 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008
Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a ChargingStatusRes message with the current SessionID, ResponseCode 'OK' and EVSENotification 'ReNegotiation' and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.4.2.2; 8.4.4.2.3; 8.8.4.1; 8.8.4.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.8.4.2.2; 8.6.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-513], [V2G2-848], [V2G2-242], [V2G2-516], [V2G2-680], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-689], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-849]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_ReceiptRequired := false, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_005	

Table 698 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **699 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009
Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a ChargingStatusRes message with the current SessionID, ResponseCode 'OK' and EVSENotification 'ReNegotiation' and all additional mandatory parameters, followed by a MeteringReceiptRes message. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.4.2.2; 8.4.4.2.3, 8.8.4.1; 8.8.4.2; 8.4.3.9.1; 8.6.2.3.2; 8.6.3.1; 8.8.4.2.2; 8.6.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-848], [V2G2-242], [V2G2-519], [V2G2-680], [V2G2-380], [V2G2-404], [V2G2-518], [V2G2-689], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-849]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_006	

Table 699 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009'.

Table **700 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PowerDeliveryRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_007	

Table 700 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **701 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_008	

Table 701 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011'.

Table **702 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a PowerDeliveryRes message with the current SessionID, , ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	
PreCondition	
f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_008	

Table 702 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **703 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013
Test objective	Test System executes GoodCase procedure and waits for a renegotiation process initiated by the SUT during the charging loop. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.4.2.2; 8.4.4.2.3; 8.8.4.1; 8.8.4.2; 8.4.3.9.1; 8.6.2.3.1; 8.6.2.3.2; 8.6.3.1; 8.8.4.2.2; 8.6.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-513], [V2G2-848], [V2G2-242], [V2G2-516], [V2G2-376], [V2G2-380], [V2G2-402], [V2G2-404], [V2G2-521], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-849]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_ReceiptRequired := false, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_009	

Table 703 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013'.

Table **704 - Test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014'**

TC Id	TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014
Test objective	Test System executes GoodCase procedure and waits for a renegotiation process initiated by the SUT during the charging loop. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters after receiving a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'ReNegotiation' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.4.2.2; 8.4.4.2.3, 8.8.4.1; 8.8.4.2; 8.4.3.9.1; 8.6.2.3.2; 8.6.3.1; 8.8.4.2.2; 8.6.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-839], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-848], [V2G2-242], [V2G2-519], [V2G2-380], [V2G2-404], [V2G2-518], [V2G2-522], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-849]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_Renegotiation := false, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_010	

Table 704 lists the test case description for 'TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014'.

10.3.3.18 EVCC test cases for MeteringReceipt

10.3.3.18.1 General information

The following sub clauses include all *V2G MeteringReceipt message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.18.2 AC specific test cases

Table **705 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_001'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_001
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 705 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_001'.

Table **706 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_002'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_002	

Table 706 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **707 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_003'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_002	

Table 707 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_003'.

Table **708 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_004'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_002	

Table 708 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **709 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_005'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_002	

Table 709 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_005'.

Table **710 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_006'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_002	

Table 710 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **711 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_007'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_007
Test objective	Test System executes GoodCase procedure and sends a valid ChargingStatusRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-517], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_MeteringReceipt_001	

Table 711 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_007'.

Table **712 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_008'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_008
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 712 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **713 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_009'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_009
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 713 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_009'.

Table **714 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_010'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of MeteringReceiptRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_MeteringReceipt_002	

Table 714 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **715 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_011'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_011
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_003	

Table 715 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_011'.

Table **716 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_012'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_012
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_AC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_003	

Table 716 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **717 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_013'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_013
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-660], [V2G2-831]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 717 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_013'.

Table **718 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_014'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_014
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-660], [V2G2-831]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 718 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **719 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_015'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_015
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-660], [V2G2-831]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 719 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_015'.

Table **720 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_016'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_016
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-660], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 720 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **721 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_017'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_017
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-660], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 721 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_017'.

Table **722 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_018'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_018
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-660], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 722 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_018'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **723 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_019'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_019
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-660], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 723 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_019'.

Table **724 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_020'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_020
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-660], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 724 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_020'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **725 - Test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_021'**

TC Id	TC_EVCC_AC_VTB_MeteringReceipt_021
Test objective	Test System executes GoodCase procedure and sends a ChargingStatusRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_', ReceiptRequired 'true', SAScheduleTupleId and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.2; 8.4.2; 8.5.2.6; 8.6.2.3.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-512], [V2G2-747], [V2G2-380], [V2G2-404], [V2G2-408], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-660], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_MeteringReceipt_001	

Table 725 lists the test case description for 'TC_EVCC_AC_VTB_MeteringReceipt_021'.

10.3.3.18.3 DC specific test cases

Table **726 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_001'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_001
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 726 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **727 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_002'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_002	

Table 727 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_002'.

Table **728 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_003'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_002	

Table 728 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **729 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_004'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'..
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_002	

Table 729 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_004'.

Table **730 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_005'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_MeteringSignatureNotValid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_002	

Table 730 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **731 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_006'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a MeteringReceiptRes message with the current SessionID, ResponseCode 'FAILED_SignatureError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_002	

Table 731 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_006'.

Table **732 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_007'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_007
Test objective	Test System executes GoodCase procedure and sends a valid CurrentDemandRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-791], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_MeteringReceipt_001	

Table 732 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **733 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_008'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_008
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 733 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_008'.

Table **734 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_009'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_009
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 734 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_009'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **735 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_010'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of MeteringReceiptRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_CMN_TB_VTB_MeteringReceipt_002	

Table 735 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_010'.

Table **736 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_011'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_011
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_003	

Table 736 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_011'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **737 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_012'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_012
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_003	

Table 737 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_012'.

Table **738 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_013'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_013
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-831]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 738 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_013'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **739 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_014'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_014
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-831]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 739 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_014'.

Table **740 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_015'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_015
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-831]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 740 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_015'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **741 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_016'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_016
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 741 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_016'.

Table **742 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_017'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_017
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 742 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_017'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **743 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_018'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_018
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_MeterReading := unknown, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 743 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_018'.

Table **744 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_019'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_019
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 744 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_019'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **745 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_020'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_020
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.10.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-891], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certUpdate, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 745 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_020'.

Table **746 - Test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_021'**

TC Id	TC_EVCC_DC_VTB_MeteringReceipt_021
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', ReceiptRequired 'true' and all additional mandatory parameters. Test System then checks that the SUT sends a correct MeteringReceiptReq message with a valid signature, current sessionID, valid SAScheduleTupleId, a MeterInfo including MeterReading and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.13.2; 8.8.4.2.3; 8.5.2.6; 8.4.2; 8.6.2.4.2; 8.6.3.1; 8.6.3.2; 8.6.2.1; 7.3.2; 7.9.2.4.2; 7.9.2.4.5; 8.4.3.11.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-245], [V2G2-246], [V2G2-276], [V2G2-277], [V2G2-790], [V2G2-747], [V2G2-394], [V2G2-405], [V2G2-789], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-005], [V2G2-006], [V2G2-007], [V2G2-117], [V2G2-119], [V2G2-652], [V2G2-764], [V2G2-765], [V2G2-766], [V2G2-767], [V2G2-768], [V2G2-769], [V2G2-770], [V2G2-771], [V2G2-903], [V2G2-909], [V2G2-896], [V2G2-831], [V2G2-776]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_CertificateHandling := certInstall, PIXIT_CMN_CMN_MeterReading := meterReading, PIXIT_EVCC_CMN_SAScheduleTupleIDMR := sAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_MeteringReceipt_001	

Table 746 lists the test case description for 'TC_EVCC_DC_VTB_MeteringReceipt_021'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.19 EVCC test cases for CableCheck

10.3.3.19.1 General information

The following sub clauses include all *V2G CableCheck message* test case descriptions, where the SUT is the EVCC.

10.3.3.19.2 DC specific test cases

Table **747 - Test case description for 'TC_EVCC_DC_VTB_CableCheck_001'**

TC Id	TC_EVCC_DC_VTB_CableCheck_001
Test objective	Test System executes GoodCase procedure and sends a ChargeParameterDiscoveryRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished', a valid scheduleList and all additional mandatory parameters. Test System then checks that the SUT applies CP State C or D and sends a correct CableCheckReq message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.5.2.2, 8.5.4.2, 8.7.3.1, 8.7.3.6, 8.7.4.4, 8.8.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1
Referenced requirement(s)	[V2G2-249], [V2G2-250], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-912], [V2G2-599], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_CMN_PR_ChargeParameterDiscovery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_001	

Table 747 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_001'.

Table **748 - Test case description for 'TC_EVCC_DC_VTB_CableCheck_002'**

TC Id	TC_EVCC_DC_VTB_CableCheck_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CableCheckRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-524], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_002	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_002	

Table 748 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **749 - Test case description for 'TC_EVCC_DC_VTB_CableCheck_003'**

TC Id	TC_EVCC_DC_VTB_CableCheck_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CableCheckRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-524], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_002	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_002	

Table 749 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_003'.

Table **750 - Test case description for 'TC_EVCC_DC_VTB_CableCheck_004'**

TC Id	TC_EVCC_DC_VTB_CableCheck_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CableCheckRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-524], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_002	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_002	

Table 750 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **751 - Test case description for 'TC_EVCC_DC_VTB_CableCheck_005'**

TC Id	TC_EVCC_DC_VTB_CableCheck_005
Test objective	Test System executes GoodCase procedure, sends a valid ChargeParameterDiscoveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a CableCheckReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-524], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_002	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_003	

Table 751 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_005'.

Table **752 - Test case description for 'TC_EVCC_DC_VTB_CableCheck_006'**

TC Id	TC_EVCC_DC_VTB_CableCheck_006
Test objective	Test System executes GoodCase procedure, sends continously a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Ongoing', EVSEStatusCode 'EVSE_IsolationMonitoringActive' and all additional mandatory parameters and waits for another CableCheckReq message until the V2G_EVCC_CableCheck_Timer is equal or larger than V2G_EVCC_CableCheck_Timeout. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.3.1; 8.7.3.6; 8.8.4.2.3
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-605], [V2G2-700], [V2G2-701], [V2G2-702], [V2G2-617]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_002	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_004	

Table 752 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **753 - Test case description for 'TC_EVCC_DC_VTB_CableCheck_007'**

TC Id	TC_EVCC_DC_VTB_CableCheck_007
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of CableCheckRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_002	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_005	

Table 753 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_007'.

Table **754 - Test case description for 'TC_EVCC_DC_VTB_CableCheck_008'**

TC Id	TC_EVCC_DC_VTB_CableCheck_008
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_002	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_006	

Table 754 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **755 - Test case description for 'TC_EVCC_DC_VTB_CableCheck_009'**

TC Id	TC_EVCC_DC_VTB_CableCheck_009
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_002	
Expected behavior	
f_EVCC_DC_TB_VTB_CableCheck_006	

Table 755 lists the test case description for 'TC_EVCC_DC_VTB_CableCheck_009'.

10.3.3.20 EVCC test cases for PreCharge

10.3.3.20.1 General information

The following sub clause includes all *V2G PreCharge message* test case descriptions, where the SUT is the EVCC.

10.3.3.20.2 DC specific test cases

Table **756 - Test case description for 'TC_EVCC_DC_VTB_PreCharge_001'**

TC Id	TC_EVCC_DC_VTB_PreCharge_001
Test objective	Test System executes GoodCase procedure and sends a CableCheckRes message with the current SessionID, ResponseCode 'OK', EVSEProcessing 'Finished' and all additional mandatory parameters. Test System then checks that the SUT sends a correct PreChargeReq message with the current SessionID, target voltage and current and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.5.4; 8.7.3.1; 8.7.3.7; 8.5.2.7; 8.4.5.3; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1
Referenced requirement(s)	[V2G2-253], [V2G2-254], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-525], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CableCheck_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_001	

Table 756 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_001'.

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **757 - Test case description for 'TC_EVCC_DC_VTB_PreCharge_002'**

TC Id	TC_EVCC_DC_VTB_PreCharge_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PreChargeRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-526], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_002	

Table 757 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_002'.

Table **758 - Test case description for 'TC_EVCC_DC_VTB_PreCharge_003'**

TC Id	TC_EVCC_DC_VTB_PreCharge_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PreChargeRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-526], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_002	

Table 758 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **759 - Test case description for 'TC_EVCC_DC_VTB_PreCharge_004'**

TC Id	TC_EVCC_DC_VTB_PreCharge_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PreChargeRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-526], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_002	

Table 759 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_004'.

Table **760 - Test case description for 'TC_EVCC_DC_VTB_PreCharge_005'**

TC Id	TC_EVCC_DC_VTB_PreCharge_005
Test objective	Test System executes GoodCase procedure, sends a valid CableCheckRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PreChargeReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-526], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_003	

Table 760 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **761 - Test case description for 'TC_EVCC_DC_VTB_PreCharge_006'**

TC Id	TC_EVCC_DC_VTB_PreCharge_006
Test objective	Test System executes GoodCase procedure and sends continously a PreChargeRes message with the current SessionID, ResponseCode 'OK', EVSEPresentVoltage '0' and all additional mandatory parameters and waits for another PreChargeReq message with the current SessionID, target voltage and current and all additional mandatory parameters until the V2G_EVCC_PreCharge_Timer is equal or larger than V2G_EVCC_PreCharge_Timeout. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.3.1; 8.7.3.7; 8.8.4.2.3
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-605], [V2G2-704], [V2G2-705], [V2G2-706], [V2G2-832], [V2G2-618]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_004	

Table 761 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_006'.

Table **762 - Test case description for 'TC_EVCC_DC_VTB_PreCharge_007'**

TC Id	TC_EVCC_DC_VTB_PreCharge_007
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PreChargeRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_005	

Table 762 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **763 - Test case description for 'TC_EVCC_DC_VTB_PreCharge_008'**

TC Id	TC_EVCC_DC_VTB_PreCharge_008
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' present voltage and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_006	

Table 763 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_008'.

Table **764 - Test case description for 'TC_EVCC_DC_VTB_PreCharge_009'**

TC Id	TC_EVCC_DC_VTB_PreCharge_009
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreCharge_006	

Table 764 lists the test case description for 'TC_EVCC_DC_VTB_PreCharge_009'.

10.3.3.21 EVCC test cases for PreChargeOrPowerDelivery

10.3.3.21.1 General information

The following sub clause includes all *V2G* PreChargeOrPowerDelivery *message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

10.3.3.21.2 DC specific test cases

Table **765 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001
Test objective	Test System executes GoodCase procedure and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.4.5.3; 8.4.3.9.1; 8.5.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.5.2.10; 8.5.2.11; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001	

Table 765 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001'.

Table **766 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 766 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **767 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 767 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003'.

Table **768 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 768 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **769 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ChargingProfileInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 769 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005'.

Table **770 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_TariffSelectionInvalid' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 770 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **771 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_PowerDeliveryNotApplied' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 771 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007'.

Table **772 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ContactorError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002	

Table 772 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **773 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009
Test objective	Test System executes GoodCase procedure, sends a valid PreChargeRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_003	

Table 773 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009'.

Table **774 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PowerDeliveryRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_004	

Table 774 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **775 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_005	

Table 775 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011'.

Table **776 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_005	

Table 776 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **777 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.4.5.3; 8.4.3.9.1; 8.5.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 777 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013'.

Table **778 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.4.5.3; 8.4.3.9.1; 8.5.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 778 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **779 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015
Test objective	Test System executes GoodCase procedure and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent before). Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.4.5.3; 8.4.3.9.1; 8.5.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.5.2.10; 8.5.2.11; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-296], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001	

Table 779 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015'.

Table **780 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016
Test objective	Test System executes GoodCase procedure and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent before). Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.4.5.3; 8.4.3.9.1; 8.5.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.5.2.10; 8.5.2.11; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_CMN_ChargingProfileOptimization := none_
PreCondition	
f_EVCC_DC_PR_PreCharge_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001	

Table 780 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **781 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent within renegotiation before). Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.4.5.3; 8.4.3.9.1; 8.5.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-296], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 781 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **782 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent within renegotiation before). Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', an optimized charging profile including the SAScheduleTuple of the sales tariff with the lowest price level and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.4.5.3; 8.4.3.9.1; 8.5.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-296], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_CMN_ChargingProfileOptimization := optimization, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 782 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **783 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent within renegotiation before). Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.4.5.3; 8.4.3.9.1; 8.5.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := none_, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 783 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **784 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired (A valid ChargeParameterDiscoveryRes message with a valid scheduleList including three SAScheduleTuples with three different price levels within the corresponding sales tariffs was sent within renegotiation before). Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', a charging profile including the default SAScheduleTuple and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.4.5.3; 8.4.3.9.1; 8.5.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.5.2.12; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-297], [V2G2-298], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_CMN_ChargingProfileOptimization := none_, PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 784 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020'.

Table **785 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021
Test objective	Test System executes GoodCase procedure, initiates a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.4.5.3; 8.4.3.9.1; 8.5.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := oldSAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 785 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **786 - Test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022'**

TC Id	TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022
Test objective	Test System executes GoodCase procedure, waits for a renegotiation process and sends a PreChargeRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. If another PreChargeReq message is received, Test System then sends a new valid PreChargeRes message until the PreCharge process has finished or the 'PreCharge_Timeout' has expired. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Start_', the previous charging profile including the same SAScheduleTupleID than before renegotiation and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.3.9.2, 8.4.3.9.3; 8.8.4.2.3; 8.4.5.3; 8.4.3.9.1; 8.5.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.5.2.10; 8.5.2.11; 8.7.4.2; 8.8.4.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-836], [V2G2-253], [V2G2-254], [V2G2-705], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-278], [V2G2-279], [V2G2-286], [V2G2-528], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-284], [V2G2-606], [V2G2-288], [V2G2-607], [V2G2-842], [V2G2-293], [V2G2-675], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown, PIXIT_EVCC_CMN_OldSAScheduleTupleID := oldSAScheduleTupleID
PreCondition	
f_EVCC_DC_PR_PreChargeRenegotiation_001	
Expected behavior	
f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006	

Table 786 lists the test case description for 'TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022'.

10.3.3.22 EVCC test cases for CurrentDemand

10.3.3.22.1 General information

The following sub clause includes all *V2G CurrentDemand message* test case descriptions, where the SUT is the EVCC.

10.3.3.22.2 DC specific test cases

Table **787 - Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_001'**

TC Id	TC_EVCC_DC_VTB_CurrentDemand_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', and all additional mandatory parameters. Test System then checks that the SUT sends a correct CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.5.4.2; 8.4.5.4.3 ;8.5.4.2; 8.4.2; 8.4.4.2; 8.5.2.7; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-257], [V2G2-258], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-530], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_PreChargeOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_001	

Table 787 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **788 - Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_002'**

TC Id	TC_EVCC_DC_VTB_CurrentDemand_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CurrentDemandRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.8.4.2.3
Referenced requirement(s)	[V2G2-532], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_002	

Table 788 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_002'.

Table **789 - Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_003'**

TC Id	TC_EVCC_DC_VTB_CurrentDemand_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CurrentDemandRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.8.4.2.3
Referenced requirement(s)	[V2G2-532], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_002	

Table 789 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **790 - Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_004'**

TC Id	TC_EVCC_DC_VTB_CurrentDemand_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a CurrentDemandRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.8.4.2.3
Referenced requirement(s)	[V2G2-532], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_002	

Table 790 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_004'.

Table **791 - Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_005'**

TC Id	TC_EVCC_DC_VTB_CurrentDemand_005
Test objective	Test System executes GoodCase procedure, sends valid PowerDeliveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a CurrentDemandReq message with the current SessionID, target voltage and current and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.8.4.2.3; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-532], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_003	

Table 791 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **792 - Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_006'**

TC Id	TC_EVCC_DC_VTB_CurrentDemand_006
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.1s and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_004	

Table 792 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_006'.

Table **793 - Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_007'**

TC Id	TC_EVCC_DC_VTB_CurrentDemand_007
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.1s and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_004	

Table 793 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **794 - Test case description for 'TC_EVCC_DC_VTB_CurrentDemand_008'**

TC Id	TC_EVCC_DC_VTB_CurrentDemand_008
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of CurrentDemandRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_CurrentDemand_005	

Table 794 lists the test case description for 'TC_EVCC_DC_VTB_CurrentDemand_008'.

10.3.3.23 EVCC test cases for ChargingLoopDcOrPowerDelivery

10.3.3.23.1 General information

The following sub clause includes all *V2G ChargingLoopDcOrPowerDelivery message* test case descriptions, where the SUT is the EVCC.

10.3.3.23.2 DC specific test cases

Table **795 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. This process will be continued for 'PICS_CMN_CMN_LoopCounter' message sequences if a new valid CurrentDemandReq message was received afterwards. In the last sequence EVSENotification will be changed to 'StopCharging'. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.4.5.4.3; 8.5.4.2; 8.4.2; 8.4.3.9.1; 8.8.4.1; 8.7.4.2; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-839], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-527], [V2G2-531], [V2G2-679], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-832], [V2G2-293], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_ReceiptRequired := false
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_001	

Table 795 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **796 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002
Test objective	Test System executes GoodCase procedure and sends a CurrentDemandRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Furthermore Test System sends a MeteringReceiptRes message with the current SessionID, ResponseCode 'Ok' and all additional mandatory parameters if a valid MeteringReceiptReq message was received before. This process will be continued for 'PICS_CMN_CMN_LoopCounter' message sequences if a new valid CurrentDemandReq message was received afterwards. In the last charging loop sequence EVSENotification will be changed to 'StopCharging'. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.4.5.4.3; 8.5.4.2; 8.4.2; 8.4.3.9.1; 8.8.4.1; ; 8.7.4.2; 8.5.4.5; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-839], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-793], [V2G2-679], [V2G2-394], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-792], [V2G2-832], [V2G2-903], [V2G2-293], [V2G2-673]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_002	

Table 796 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002'.

Table **797 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003	

Table 797 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **798 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003	

Table 798 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004'.

Table **799 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003	

Table 799 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **800 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a PowerDeliveryRes message with the current SessionID, ResponseCode 'FAILED_ContactorError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003	

Table 800 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006'.

Table **801 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007
Test objective	Test System executes GoodCase procedure, sends a valid CurrentDemandRes or MeteringReceiptRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a PowerDeliveryReq message with the current SessionID, ChargeProgress 'Stop_' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-799], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_004	

Table 801 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **802 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008
Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'Renegotiation' and all additional mandatory parameters. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.4.5.4.3; 8.5.4.2; 8.4.2; 8.4.3.9.1; 8.8.4.1; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-531], [V2G2-680], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-832], [V2G2-686]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_ReceiptRequired := false, PICS_CMN_CMN_Renegotiation := true
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_005	

Table 802 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008'.

Table **803 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009
Test objective	Test System executes GoodCase procedure and initiates a renegotiation by sending a CurrentDemandRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'Renegotiation' and all additional mandatory parameters, followed by a MeteringReceiptRes message. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.4.5.4.3; 8.5.4.2; 8.4.2; 8.4.3.9.1; 8.8.4.1; 8.5.4.5; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.8.4.2.3; 8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369],[V2G2-680], [V2G2-394], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-792], [V2G2-840], [V2G2-841], [V2G2-832], [V2G2-794]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_Renegotiation := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := unknown, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_006	

Table 803 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **804 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of PowerDeliveryRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_007	

Table 804 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010'.

Table **805 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_008	

Table 805 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **806 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'None_' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_008	

Table 806 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012'.

Table **807 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013
Test objective	Test System executes GoodCase procedure and waits for a renegotiation process initiated by the SUT during the charging loop. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.4.5.4.3; 8.5.4.2; 8.4.2; 8.4.3.9.1; 8.8.4.1; 8.5.4.5; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-531], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-840], [V2G2-841], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_ReceiptRequired := false, PICS_CMN_CMN_Renegotiation := false
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_009	

Table 807 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **808 - Test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014'**

TC Id	TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014
Test objective	Test System executes GoodCase procedure and waits for a renegotiation process initiated by the SUT during the charging loop. Test System then checks that the SUT sends a correct PowerDeliveryReq message with the current SessionID, ChargeProgress 'Renegotiate' and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.9.2, 8.4.3.9.3; 8.4.5.4.3; 8.5.4.2; 8.4.2; 8.4.3.9.1; 8.8.4.1; 8.5.4.5; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.8.4.2.3; 8.7.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-182], [V2G2-183], [V2G2-221], [V2G2-222], [V2G2-374], [V2G2-375], [V2G2-747], [V2G2-278], [V2G2-279], [V2G2-257], [V2G2-258], [V2G2-286], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-394], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-792], [V2G2-840], [V2G2-841], [V2G2-832]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_Renegotiation := false, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_Renegotiation := renegotiation, PIXIT_CMN_CMN_PkiName := PKI_1
PreCondition	
f_EVCC_DC_PR_MeteringReceipt_001	
Expected behavior	
f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_010	

Table 808 lists the test case description for 'TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014'.

10.3.3.24 EVCC test cases for WeldingDetection

10.3.3.24.1 General information

The following sub clause includes all *V2G WeldingDetection message* test case descriptions, where the SUT is the EVCC.

10.3.3.24.2 DC specific test cases

Table **809 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_001'**

TC Id	TC_EVCC_DC_VTB_WeldingDetection_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B and sends a correct WeldingDetectionReq message with the current SessionID and all additional mandatory parameters.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.5.5.2; 8.8.4.2.3; 8.4.2; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-261], [V2G2-262], [V2G2-533], [V2G2-747], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_001	

Table 809 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **810 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_002'**

TC Id	TC_EVCC_DC_VTB_WeldingDetection_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a WeldingDetectionRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-534], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_002	

Table 810 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_002'.

Table **811 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_003'**

TC Id	TC_EVCC_DC_VTB_WeldingDetection_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a WeldingDetectionRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-534], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_002	

Table 811 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **812 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_004'**

TC Id	TC_EVCC_DC_VTB_WeldingDetection_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a WeldingDetectionRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1;
Referenced requirement(s)	[V2G2-534], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_002	

Table 812 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_004'.

Table **813 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_005'**

TC Id	TC_EVCC_DC_VTB_WeldingDetection_005
Test objective	Test System executes GoodCase procedure, sends a valid PowerDeliveryRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving a WeldingDetectionReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-534], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_003	

Table 813 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_005'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **814 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_006'**

TC Id	TC_EVCC_DC_VTB_WeldingDetection_006
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of WeldingDetectionRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_004	

Table 814 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_006'.

Table **815 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_007'**

TC Id	TC_EVCC_DC_VTB_WeldingDetection_007
Test objective	Test System executes GoodCase procedure, signals CP State E, waits for 0.5s and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State E is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_005	

Table 815 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **816 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_008'**

TC Id	TC_EVCC_DC_VTB_WeldingDetection_008
Test objective	Test System executes GoodCase procedure, signals CP State F, waits for 0.5s and sends a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters. Test System then checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' as soon as CP State F is applied. If the TCP connection is closed before the corresponding request message is sent by the Test System, the send operation shall be canceled and the test result shall be interpreted as successful.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.4.2
Referenced requirement(s)	[V2G2-728], [V2G2-025], [V2G2-737]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection
PreCondition	
f_EVCC_DC_PR_WeldingDetection_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_005	

Table 816 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_008'.

Table **817 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetection_009'**

TC Id	TC_EVCC_DC_VTB_WeldingDetection_009
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B and sends a correct WeldingDetectionReq message with the current SessionID and all additional mandatory parameters. Furthermore the repetition of the WeldingDetection message sequence will be checked by sending a WeldingDetectionRes message with the current SessionID, ResponseCode 'OK', EVSEPresentVoltage > 60V and all additional mandatory parameters after receipt of the first WeldingdetectionReq message.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.5.5.2; 8.8.4.2.3; 8.4.2; 8.5.4.2; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4; 8.8.4.2.3
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-261], [V2G2-262], [V2G2-533], [V2G2-747], [V2G2-367], [V2G2-368], [V2G2-369], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-620]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := weldingDetection
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetection_006	

Table 817 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetection_009'.

10.3.3.25 EVCC test cases for WeldingDetectionOrSessionStop

10.3.3.25.1 General information

The following sub clause includes all *V2G WeldingDetectionOrSessionStop message* test case descriptions, where the SUT is the EVCC.

© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

10.3.3.25.2 DC specific test cases

Table **818 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-660], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 818 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001'.

Table **819 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004	

Table 819 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **820 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED_SequenceError' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004	

Table 820 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003'.

Table **821 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004
Test objective	Test System executes GoodCase procedure and turns off the oscillator before sending a SessionStopRes message with the current SessionID, ResponseCode 'FAILED_UnknownSession' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004	

Table 821 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **822 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005
Test objective	Test System executes GoodCase procedure, sends a valid sends a PowerDeliveryRes or WeldingDetectionRes message and waits until the 'V2G_Msg_Timeout' timer has expired after receiving the SessionStopReq message with the current SessionID and all additional mandatory parameters. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.8.4.2.1; 8.7.2.1; 8.7.2.2
Referenced requirement(s)	[V2G2-507], [V2G2-728], [V2G2-025], [V2G2-434], [V2G2-436], [V2G2-437], [V2G2-438]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_005	

Table 822 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005'.

Table **823 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006
Test objective	Test System executes GoodCase procedure and sends a valid PaymentServiceSelectionRes message instead of SessionStopRes message. Test System then checks that the SUT applies CP State B within 'par_EVCC_StateB_Shutdown_Timeout' (if not already applied) and terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout'.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 7.4; 8.2.1; 8.8.4.2.1
Referenced requirement(s)	[V2G2-482], [V2G2-728], [V2G2-025]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	
PreCondition	
f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_006	

Table 823 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **824 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.6.2.1; 8.7.4.4; 7.3.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-633], [V2G2-635], [V2G2-637]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 824 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007'.

Table **825 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If the SessionStopReq message contains ChargingSession 'Terminate', the termination of the data link connection within 'par_T_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.6.2.1; 8.7.4.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 825 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **826 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 826 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009'.

Table **827 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.6.2.1; 8.7.4.4; 7.3.4; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := none , PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 827 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **828 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Terminate' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. Afterwards the termination of the data link connection within 'par_T_match_leave' will be checked. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.6.2.1; 8.7.4.4; 7.4
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-717]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 828 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011'.

Table **829 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := tls, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 829 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **830 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.6.2.1; 8.7.4.4; 7.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := none_, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 830 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013'.

Table **831 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters. Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := unknown, PIXIT_EVCC_CMN_Pause := pause
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 831 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **832 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 832 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015'.

Table **833 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.6.2.1; 8.7.4.4; 7.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 833 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **834 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the first Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepWithoutCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 834 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017'.

Table **835 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.4.3.12; 8.4.5.5; 8.4.1; 8.8.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.2.4.2; 8.6.3.1; 8.6.2.1; 8.7.4.4; 8.3.2; 8.3.3; 8.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-394], [V2G2-403], [V2G2-405], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := tls
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 835 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **836 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.6.2.1; 8.7.4.4; 7.3.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-633], [V2G2-635], [V2G2-637], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := none_
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 836 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019'.

Table **837 - Test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020'**

TC Id	TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020
Test objective	Test System executes GoodCase procedure and sends a PowerDeliveryRes message with the current SessionID, ResponseCode 'OK', EVSENotification 'StopCharging' and all additional mandatory parameters (Test System has sent a PmaxScheduleList with the second Pmax element = 0W (par_SECC_Pmax0W) within ChargeParameterDiscoveryRes message before). Test System then checks that the SUT applies CP State B, sends a correct SessionStopReq message with the current SessionID, ChargingSession 'Pause' and all additional mandatory parameters. Furthermore Test System checks that the SUT terminates the V2G communication session by closing the TCP connection within 'par_CMN_TCP_Connection_Termination_Timeout' if a SessionStopRes message with the current SessionID, ResponseCode 'OK' and all additional mandatory parameters was received before. If a valid WeldingDetectionReq message was received before, the corresponding message sequence shall be executed by SUT and Test System previously.
Document reference	Document: ISO:15118-2:2014:IS Section(s): 8.3.2; 8.3.3; 8.3.4; 8.4.2; 8.4.3.12.1; 8.8.4.2.3; 8.6.2.4.1; 8.6.3.1; 8.6.2.1; 8.7.4.4; 7.4; 8.4.2
Referenced requirement(s)	[V2G2-179], [V2G2-180], [V2G2-181], [V2G2-183], [V2G2-747], [V2G2-239], [V2G2-738], [V2G2-535], [V2G2-390], [V2G2-403], [V2G2-659], [V2G2-661], [V2G2-662], [V2G2-913], [V2G2-914], [V2G2-660], [V2G2-718], [V2G2-739]
Config Id	CF_04_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_CMN_CMN_IdentificationMode := eIM, PICS_EVCC_CMN_PmaxSchedulewithZeroPow := sleepAfterCharge
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_TLS := unknown
PreCondition	
f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001	
Expected behavior	
f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 837 lists the test case description for 'TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

11 Reliability and Robustness Testing

11.1 General information

The specified abstact test cases in section 8, 9 and 10 do not include the assessment of performance nor robustness or reliability of an implementation. But reliability and robustness of a conforming implementation is also important for the successful operation of charging infrastructures.

The robustness is the ability of a charging system to cope with errors during execution. Reliability is the ability of a system to function with given environmental conditions, for an amount of time.

This clause covers all additional test case descriptions for reliability and robustness testing.

11.2 SECC test cases

11.2.1 General information

This sub clause covers all reliability test case descriptions, where the SUT is the SECC.

11.2.2 AC specific test cases

Table **838 - Test case description for 'TC_SECC_AC_VTB_Reliability_001'**

TC Id	`TC_SECC_AC_VTB_Reliability_001`
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the AC charging process for 2 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 20 times. Test System checks that the SUT is able to charge for 2 min. Furthermore, Test System checks that this process is successful for 20 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_001
PICS selection	
PIXIT selection	
PreCondition	
`f_SECC_CMN_PR_Authorization_001`	
Expected behavior	
`f_SECC_AC_TB_VTB_Reliability_001, f_SECC_AC_TB_VTB_PowerDelivery_001, f_SECC_CMN_TB_VTB_SessionStop_001`	

Table 838 lists the test case description for 'TC_SECC_AC_VTB_Reliability_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **839 - Test case description for 'TC_SECC_AC_VTB_Reliability_002'**

TC Id	TC_SECC_AC_VTB_Reliability_002
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the AC charging process for 15 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 2 times. Test System checks that the SUT is able to charge for 15 min. Furthermore, Test System checks that this process is successful for 2 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_Reliability_001, f_SECC_AC_TB_VTB_PowerDelivery_001, f_SECC_CMN_TB_VTB_SessionStop_001	

Table 839 lists the test case description for 'TC_SECC_AC_VTB_Reliability_002'.

Table **840 - Test case description for 'TC_SECC_AC_VTB_Reliability_003'**

TC Id	TC_SECC_AC_VTB_Reliability_003
Test objective	Test System executes GoodCase procedure and enters the AC charging process for 45 min. Afterwards, Test System stops the charging process for a non-critical reason and initiates a plug-out. Test System checks that the SUT is able to charge for 45 min.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_001
PICS selection	
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_AC_TB_VTB_Reliability_001, f_SECC_AC_TB_VTB_PowerDelivery_001, f_SECC_CMN_TB_VTB_SessionStop_001	

Table 840 lists the test case description for 'TC_SECC_AC_VTB_Reliability_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

11.2.3 DC specific test cases

Table **841 - Test case description for 'TC_SECC_DC_VTB_Reliability_001'**

TC Id	TC_SECC_DC_VTB_Reliability_001
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the DC charging process for 2 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 20 times. Test System checks that the SUT is able to charge for 2 min. Furthermore, Test System checks that this process is successful for 20 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_001
PICS selection	PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_Reliability_001, f_SECC_DC_TB_VTB_PowerDelivery_001, f_SECC_DC_TB_VTB_WeldingDetection_001, f_SECC_CMN_TB_VTB_SessionStop_001	

Table 841 lists the test case description for 'TC_SECC_DC_VTB_Reliability_001'.

Table **842 - Test case description for 'TC_SECC_DC_VTB_Reliability_002'**

TC Id	TC_SECC_DC_VTB_Reliability_002
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the DC charging process for 15 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 2 times. Test System checks that the SUT is able to charge for 15 min. Furthermore, Test System checks that this process is successful for 2 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_001
PICS selection	PICS_SECC_DC_WeldingDetection := true
PIXIT selection	
PreCondition	
f_SECC_CMN_PR_Authorization_001	
Expected behavior	
f_SECC_DC_TB_VTB_Reliability_001, f_SECC_DC_TB_VTB_PowerDelivery_001, f_SECC_DC_TB_VTB_WeldingDetection_001, f_SECC_CMN_TB_VTB_SessionStop_001	

Table 842 lists the test case description for 'TC_SECC_DC_VTB_Reliability_002'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **843 - Test case description for 'TC_SECC_DC_VTB_Reliability_003'**

TC Id	`TC_SECC_DC_VTB_Reliability_003`
Test objective	Test System executes GoodCase procedure and enters the DC charging process for 45 min. Afterwards, Test System stops the charging process for a non-critical reason and initiates a plug-out. Test System checks that the SUT is able to charge for 45 min.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_001
PICS selection	`PICS_SECC_DC_WeldingDetection := true`
PIXIT selection	
PreCondition	
`f_SECC_CMN_PR_Authorization_001`	
Expected behavior	
`f_SECC_DC_TB_VTB_Reliability_001, f_SECC_DC_TB_VTB_PowerDelivery_001, f_SECC_DC_TB_VTB_WeldingDetection_001, f_SECC_CMN_TB_VTB_SessionStop_001`	

Table 843 lists the test case description for 'TC_SECC_DC_VTB_Reliability_003'.

11.3 EVCC test cases

11.3.1 General information

This sub clause covers all reliability test case descriptions, where the SUT is the SECC.

11.3.2 AC specific test cases

Table **844 - Test case description for 'TC_EVCC_AC_VTB_Reliability_001'**

TC Id	`TC_EVCC_AC_VTB_Reliability_001`
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the AC charging process for 2 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 20 times. Test System checks that the SUT is able to charge for 2 min. Furthermore, Test System checks that this process is successful for 20 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	`PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_ReceiptRequired := false`
PIXIT selection	`PIXIT_EVCC_CMN_Pause == unknown`
PreCondition	
`f_EVCC_AC_PR_ChargingStatus_001`	
Expected behavior	
`f_EVCC_AC_TB_VTB_Reliability_001, f_EVCC_AC_TB_VTB_SessionStop_001`	

Table 844 lists the test case description for 'TC_EVCC_AC_VTB_Reliability_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **845 - Test case description for 'TC_EVCC_AC_VTB_Reliability_002'**

TC Id	TC_EVCC_AC_VTB_Reliability_002
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the AC charging process for 15 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 2 times. Test System checks that the SUT is able to charge for 15 min. Furthermore, Test System checks that this process is successful for 2 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_ReceiptRequired := false
PIXIT selection	PIXIT_EVCC_CMN_Pause == unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_Reliability_001, f_EVCC_AC_TB_VTB_SessionStop_001	

Table 845 lists the test case description for 'TC_EVCC_AC_VTB_Reliability_002'.

Table **846 - Test case description for 'TC_EVCC_AC_VTB_Reliability_003'**

TC Id	TC_EVCC_AC_VTB_Reliability_003
Test objective	Test System executes GoodCase procedure and enters the AC charging process for 45 min. Afterwards, Test System stops the charging process for a non-critical reason and initiates a plug-out. Test System checks that the SUT is able to charge for 45 min.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	PICS_CMN_CMN_ChargingMode := aC, PICS_EVCC_CMN_ReceiptRequired := false
PIXIT selection	PIXIT_EVCC_CMN_Pause == unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_Reliability_001, f_EVCC_AC_TB_VTB_SessionStop_001	

Table 846 lists the test case description for 'TC_EVCC_AC_VTB_Reliability_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **847 - Test case description for 'TC_EVCC_AC_VTB_Reliability_005'**

TC Id	TC_EVCC_AC_VTB_Reliability_005
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the AC charging process for 2 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 20 times. Test System checks that the SUT is able to charge for 2 min. Furthermore, Test System checks that this process is successful for 20 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause == unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_Reliability_002, f_EVCC_AC_TB_VTB_SessionStop_001	

Table 847 lists the test case description for 'TC_EVCC_AC_VTB_Reliability_005'.

Table **848 - Test case description for 'TC_EVCC_AC_VTB_Reliability_006'**

TC Id	TC_EVCC_AC_VTB_Reliability_006
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the AC charging process for 15 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 2 times. Test System checks that the SUT is able to charge for 15 min. Furthermore, Test System checks that this process is successful for 2 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause == unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_Reliability_002, f_EVCC_AC_TB_VTB_SessionStop_001	

Table 848 lists the test case description for 'TC_EVCC_AC_VTB_Reliability_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **849 - Test case description for 'TC_EVCC_AC_VTB_Reliability_007'**

TC Id	TC_EVCC_AC_VTB_Reliability_007
Test objective	Test System executes GoodCase procedure and enters the AC charging process for 45 min. Afterwards, Test System stops the charging process for a non-critical reason and initiates a plug-out. Test System checks that the SUT is able to charge for 45 min.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := aC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_CMN_Pause == unknown
PreCondition	
f_EVCC_AC_PR_ChargingStatus_001	
Expected behavior	
f_EVCC_AC_TB_VTB_Reliability_002, f_EVCC_AC_TB_VTB_SessionStop_001	

Table 849 lists the test case description for 'TC_EVCC_AC_VTB_Reliability_007'.

11.3.3 DC specific test cases

Table **850 - Test case description for 'TC_EVCC_DC_VTB_Reliability_001'**

TC Id	TC_EVCC_DC_VTB_Reliability_001
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the DC charging process for 2 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 20 times. Test System checks that the SUT is able to charge for 2 min. Furthermore, Test System checks that this process is successful for 20 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_ReceiptRequired := false
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_Pause == unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_Reliability_001, f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 850 lists the test case description for 'TC_EVCC_DC_VTB_Reliability_001'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Table **851 - Test case description for 'TC_EVCC_DC_VTB_Reliability_002'**

TC Id	TC_EVCC_DC_VTB_Reliability_002
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the DC charging process for 15 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 2 times. Test System checks that the SUT is able to charge for 15 min. Furthermore, Test System checks that this process is successful for 2 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_ReceiptRequired := false
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_Pause == unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_Reliability_001, f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 851 lists the test case description for 'TC_EVCC_DC_VTB_Reliability_002'.

Table **852 - Test case description for 'TC_EVCC_DC_VTB_Reliability_003'**

TC Id	TC_EVCC_DC_VTB_Reliability_003
Test objective	Test System executes GoodCase procedure and enters the DC charging process for 45 min. Afterwards, Test System stops the charging process for a non-critical reason and initiates a plug-out. Test System checks that the SUT is able to charge for 45 min.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	PICS_CMN_CMN_ChargingMode := dC, PICS_EVCC_CMN_ReceiptRequired := false
PIXIT selection	PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_Pause == unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_Reliability_001, f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 852 lists the test case description for 'TC_EVCC_DC_VTB_Reliability_003'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **853 - Test case description for 'TC_EVCC_DC_VTB_Reliability_005'**

TC Id	TC_EVCC_DC_VTB_Reliability_005
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the DC charging process for 2 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 20 times. Test System checks that the SUT is able to charge for 2 min. Furthermore, Test System checks that this process is successful for 20 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_Pause == unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_Reliability_002, f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 853 lists the test case description for 'TC_EVCC_DC_VTB_Reliability_005'.

Table **854 - Test case description for 'TC_EVCC_DC_VTB_Reliability_006'**

TC Id	TC_EVCC_DC_VTB_Reliability_006
Test objective	Test System initiates a plug-in, executes GoodCase procedure and enters the DC charging process for 15 min. Afterwards, Test System stops the charging process for a non-critical reason, initiates a plug-out and waits for 20s. This process will be executed 2 times. Test System checks that the SUT is able to charge for 15 min. Furthermore, Test System checks that this process is successful for 2 times in a row.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_Pause == unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_Reliability_002, f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 854 lists the test case description for 'TC_EVCC_DC_VTB_Reliability_006'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Table **855 - Test case description for 'TC_EVCC_DC_VTB_Reliability_007'**

TC Id	TC_EVCC_DC_VTB_Reliability_007
Test objective	Test System executes GoodCase procedure and enters the DC charging process for 45 min. Afterwards, Test System stops the charging process for a non-critical reason and initiates a plug-out. Test System checks that the SUT is able to charge for 45 min.
Document reference	Document: Section(s):
Referenced requirement(s)	
Config Id	CF_02_002
PICS selection	PICS_CMN_CMN_IdentificationMode := pnC, PICS_EVCC_CMN_ReceiptRequired := true, PICS_CMN_CMN_ChargingMode := dC
PIXIT selection	PIXIT_CMN_CMN_PkiName := PKI_1, PIXIT_EVCC_DC_WeldingDetection := unknown, PIXIT_EVCC_CMN_Pause == unknown
PreCondition	
f_EVCC_DC_PR_CurrentDemand_001	
Expected behavior	
f_EVCC_DC_TB_VTB_Reliability_002, f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001	

Table 855 lists the test case description for 'TC_EVCC_DC_VTB_Reliability_007'.

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex A
(normative)

Configuration specifications

A.1 General information

This annex covers all test configuration related information in the ATS.

A.2 Parameter configuration

```
module ParameterConfiguration {

    import from Services_HAL_61851_1 all;
    import from urn_iso_15118_2_2010_AppProtocol language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from XSDAUX all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from Pics_15118_2 all;
    import from Pics_15118 all;
    import from DataStructure_HAL_61851 all;
    import from DataStructure_V2GTP all;
    import from DataStructure_SDP all;
    import from DataStructure_V2G_TCP_TLS_Port_Control all;
    import from DataStructure_EIMStatus all;
    import from DataStructure_SLAC all;
    import from http_www_w3_org_2000_09_xmldsig all;
    import from urn_iso_15118_2_2013_X509Def all;
    import from DataStructure_Security all;
    import from LibFunctions_15118_3 all;
    import from DataStructure_PowerFlow all;

    //---------------Module Parameter------------
    modulepar float par_validDutyCycleLowerBound := 3.0;
    modulepar float par_validDutyCycleUpperBound := 7.0;
    modulepar float par_ignoredDutyCycleLowerBound := 0.0;
    modulepar float par_ignoredDutyCycleUpperBound := 0.0;
    modulepar float par_validFrequencyLowerBound := 980.0;
    modulepar float par_validFrequencyUpperBound := 1020.0;
    modulepar float par_validVoltageStateALowerBound := 11.0;
    modulepar float par_validVoltageStateAUpperBound := 13.0;
    modulepar float par_validVoltageStateBLowerBound := 8.0;
    modulepar float par_validVoltageStateBUpperBound := 10.0;
    modulepar float par_validVoltageStateCLowerBound := 5.0;
    modulepar float par_validVoltageStateCUpperBound := 7.0;
    modulepar float par_validVoltageStateDLowerBound := 2.0;
    modulepar float par_validVoltageStateDUpperBound := 4.0;
    modulepar float par_validVoltageStateELowerBound := -1.0;
    modulepar float par_validVoltageStateEUpperBound := 1.0;
    modulepar float par_validVoltageStateFLowerBound := -13.0;
    modulepar float par_validVoltageStateFUpperBound := -11.0;
    modulepar float par_AC_validNominalDutyCycleLowerBound := 10.0;
    modulepar float par_AC_validNominalDutyCycleUpperBound := 96.0;
    modulepar charstring par_cipherSuites := "TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256, " &
                                             "TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256";
    modulepar float par_CMN_MaxPhysicalValueTypeAmpere := 400.0;
    modulepar float par_CMN_MaxPhysicalValueTypeVolt := 1000.0;
    modulepar float par_CMN_MaxPhysicalValueTypeWattOrWatthour := 200000.0;
    modulepar float par_CMN_MaxPhysicalValueTypeSeconds := 172800.0;
    modulepar PhysicalValueType par_RemainingTimeToFullSoC := { multiplier := 0,
                                                                 unit := s,
                                                                 value_ := 3600 };
    modulepar PhysicalValueType par_RemainingTimeToBulkSoC := { multiplier := 0,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                        unit := s,
                                                        value_ := 2500 };
modulepar EnergyTransferModeType par_requestedEnergyTransferModeAC := aC_three_phase_core;
modulepar EnergyTransferModeType par_requestedEnergyTransferModeDC := dC_extended;
modulepar PhysicalValueType par_EVMaxCurrent := { multiplier := 0,
                                                  unit := a,
                                                  value_ := 32 };
modulepar PhysicalValueType par_EVMinCurrent := { multiplier := 0,
                                                  unit := a,
                                                  value_ := 0 };
modulepar PhysicalValueType par_EVMaxVoltage := { multiplier := 0,
                                                  unit := v,
                                                  value_ := 400 };
modulepar PhysicalValueType par_EVSEMaxCurrent := { multiplier := 0,
                                                    unit := a,
                                                    value_ := 32 };
modulepar PhysicalValueType par_EVSENominalVoltage := { multiplier := 0,
                                                        unit := v,
                                                        value_ := 230 };
modulepar float par_EVSENominalDutyCycle := 52.0;
modulepar PhysicalValueType par_EVMaximumCurrentLimit := { multiplier := 0,
                                                           unit := a,
                                                           value_ := 20 };
modulepar PhysicalValueType par_EVMaximumVoltageLimit := { multiplier := 0,
                                                           unit := v,
                                                           value_ := 400 };
modulepar PhysicalValueType par_EVMaximumPowerLimit := { multiplier := 0,
                                                         unit := w,
                                                         value_ := 10000 };
modulepar PhysicalValueType par_EVSEMaximumCurrentLimit := { multiplier := 0,
                                                             unit := a,
                                                             value_ := 20 };
modulepar PhysicalValueType par_EVSEMaximumVoltageLimit := { multiplier := 0,
                                                             unit := v,
                                                             value_ := 750 };
modulepar PhysicalValueType par_EVSEMaximumPowerLimit := { multiplier := 0,
                                                           unit := w,
                                                           value_ := 10000 };
modulepar PhysicalValueType par_EVSEMinimumCurrentLimit := { multiplier := 0,
                                                             unit := a,
                                                             value_ := 0 };
modulepar PhysicalValueType par_EVSEMinimumVoltageLimit := { multiplier := 0,
                                                             unit := v,
                                                             value_ := 0 };
modulepar PercentValueType par_EVRESSSOC := 10;
modulepar MACAddress_TYPE par_testSystem_mac := '000000000000'H;
modulepar MACAddress_TYPE par_testSystem_plc_node_mac := '000000000000'H;
modulepar MACAddress_TYPE par_slac_node2_mac := '021122334422'H;
modulepar MACAddress_TYPE par_slac_node3_mac := '021122334433'H;
modulepar MACAddress_TYPE par_slac_node4_mac := '021122334444'H;
modulepar MACAddress_TYPE par_slac_node5_mac := '021122334455'H;
modulepar integer par_C_EV_match_retry := 2;
modulepar SECC_IPaddress_TYPE par_SECC_IPaddress := '00000000000000000000000000000001'H;
modulepar SAIDType par_CMN_SAScheduleTuple := 1;
modulepar integer par_EVCC_NotificationMaxDelay := 2;
modulepar integer par_CMN_MaxEntriesSAScheduleTuple := 3;
modulepar integer par_C_vald_nb_toggles := 2;
modulepar integer par_SECC_attenuationDeviation := 7;
modulepar integer par_C_sequ_retry := 2;
modulepar integer par_C_EV_match_MNBC := 10;
modulepar integer par_EVCC_InsertionLossCorrection := 25;
modulepar integer par_EVCC_numberOfStartAtten := 3;
modulepar integer par_EVCC_numberOfSoundings := 10;

modulepar integer par_maxRepetitionSDP := 50;

//---------------Constants------------

const MACAddress_TYPE cc_eth_broadcast := 'FFFFFFFFFFFF'H;
const integer cc_proximity_type2_AC_13A := 1500;
const integer cc_proximity_type2_AC_20A := 680;
const integer cc_proximity_type2_AC_32A := 220;
const integer cc_proximity_type2_AC_63A := 100;
const integer cc_evse_proximity_type2_DC := 1500;
const integer cc_evse_proximity_type1 := 150;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
const integer cc_ev_proximity_type1 := 2700;
const integer cc_evse_proximity_type1EmergencySD := 480;
const float cc_offset := 1.0;
const float cc_dutyCycle_32A := 53.0;
const float cc_dutyCycle_16A := 27.0;
const MACAddress_TYPE cc_plcNode_mac_default := '00b052000001'H;
const hexstring cc_amLenValid := '0395'H;
const hexstring cc_amLenInvalid := '003A'H;
const float cc_maximum_value_physicalValue_case1 := par_CMN_MaxPhysicalValueTypeAmpere;
const float cc_maximum_value_physicalValue_case2 := par_CMN_MaxPhysicalValueTypeVolt;
const float cc_maximum_value_physicalValue_case3 := par_CMN_MaxPhysicalValueTypeSeconds;
const float cc_maximum_value_physicalValue_case4 := par_CMN_MaxPhysicalValueTypeWattOrWatthour;
const integer cc_maximum_value_meterReading := 999999999;
const hexstring cc_hexTcp := '10'H;
const hexstring cc_hexTls := '00'H;
const integer cc_chargingTime24h := 86400;
const charstring cc_eAmount := "EAmount";
const charstring cc_eVMaxVoltage := "EVMaxVoltage";
const charstring cc_eVMaxCurrent := "EVMaxCurrent";
const charstring cc_eVMinCurrent := "EVMinCurrent";
const charstring cc_eVMaximumCurrentLimit := "EVMaximumCurrentLimit";
const charstring cc_eVMaximumVoltageLimit := "EVMaximumVoltageLimit";
const charstring cc_eVMaximumPowerLimit := "EVMaximumPowerLimit";
const charstring cc_eVTargetCurrent := "EVTargetCurrent";
const charstring cc_eVTargetVoltage := "EVTargetVoltage";
const charstring cc_eVSENominalVoltage := "EVSENominalVoltage";
const charstring cc_eVSEMaxCurrent := "EVSEMaxCurrent";
const charstring cc_eVSEMaximumCurrentLimit := "EVSEMaximumCurrentLimit";
const charstring cc_eVSEMaximumVoltageLimit := "EVSEMaximumVoltageLimit";
const charstring cc_eVSEMaximumPowerLimit := "EVSEMaximumPowerLimit";
const charstring cc_eVSEMinimumVoltageLimit := "EVSEMinimumVoltageLimit";
const charstring cc_eVSEMinimumCurrentLimit := "EVSEMinimumCurrentLimit";
const charstring cc_eVSEPeakCurrentRipple := "EVSEPeakCurrentRipple";
const charstring cc_eVSEPresentVoltage := "EVSEPresentVoltage";
const charstring cc_eVSEPresentCurrent := "EVSEPresentCurrent";
const SECC_IPaddress_TYPE cc_multicast  := 'ff020000000000000000000000000001'H;
const charstring cc_crt_evse_leaf_valid := "CRT_EVSE_LEAF_VALID";
const charstring cc_crt_cpo_sub1_valid := "CRT_CPO_SUB1_VALID";
const charstring cc_crt_cpo_sub2_valid := "CRT_CPO_SUB2_VALID";
const charstring cc_crt_mo_sub1_valid := "CRT_MO_SUB1_VALID";
const charstring cc_crt_mo_sub2_valid := "CRT_MO_SUB2_VALID";
const charstring cc_crt_contract_leaf_valid := "CRT_CONTRACT_LEAF_VALID";
const charstring cc_crt_oem_leaf_valid := "CRT_OEM_LEAF_VALID";
const charstring cc_crt_v2g_root_valid := "CRT_V2G_ROOT_VALID";
const charstring cc_crt_pe_root_valid := "CRT_PE_ROOT_VALID";
const charstring cc_crt_pe_leaf_valid := "CRT_PE_LEAF_VALID";
const charstring cc_crt_oem_leaf_expired := "CRT_OEM_LEAF_EXPIRED";
const charstring cc_crt_oem_leaf_invalid_content := "CRT_OEM_LEAF_INVALID_CONTENT";
const charstring cc_crt_contract_leaf_expires := "CRT_CONTRACT_LEAF_EXPIRES";
const charstring cc_crt_contract_leaf_wrong_emaid := "CRT_CONTRACT_LEAF_WRONG_EMAID";
const charstring cc_crt_contract_leaf_invalid_emaid := "CRT_CONTRACT_LEAF_INVALID_EMAID";
const charstring cc_crt_contract_leaf_expired := "CRT_CONTRACT_LEAF_EXPIRED";
const charstring cc_crt_contract_leaf_invalid_content := "CRT_CONTRACT_LEAF_INVALID_CONTENT";
const charstring cc_crt_contract_leaf_expired_issuer_sub1 :=
                 "CRT_CONTRACT_LEAF_EXPIRED_ISSUER_SUB1";
const charstring cc_crt_contract_leaf_expired_issuer_sub2 :=
                 "CRT_CONTRACT_LEAF_EXPIRED_ISSUER_SUB2";
const charstring cc_crt_mo_sub2_expired := "CRT_MO_SUB2_EXPIRED";
const charstring cc_crt_mo_sub1_expired := "CRT_MO_SUB1_EXPIRED";
const charstring cc_crt_prof_sub1_valid := "CRT_PROV_SUB1_VALID";
const charstring cc_crt_prof_sub2_valid := "CRT_PROV_SUB2_VALID";
const charstring cc_crt_prof_leaf_valid := "CRT_PROV_LEAF_VALID";
const charstring cc_crt_prof_leaf_expired := "CRT_PROV_LEAF_EXPIRED";
const charstring cc_algorithm_method_1 := "http://www.w3.org/TR/canonical-exi/";
const charstring cc_algorithm_method_2 := "http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256";
const charstring cc_algorithm_method_3 := "http://www.w3.org/2001/04/xmlenc#sha256";
const float cc_salesTariff_processingTime_remains := 0.5;
const integer cc_preChargeVoltageRampUpSteps := 10;
const integer vc_numberOfScheduleEntries := 12;
const integer vc_maxStartValue := 16777214;

//---------------Component Variables------------

type component Tester {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
//declaration

var default vc_Default_IEC_61851_ListenerBehavior;
var default vc_Default_New_SLAC_ProcessHandling;
var float vc_validDutyCycleLowerBound1;
var float vc_validDutyCycleUpperBound1;
var float vc_validDutyCycleLowerBound2;
var float vc_validDutyCycleUpperBound2;
var hexstring vc_SECC_IPaddress;
var hexstring vc_Num_sounds;
var hexstring vc_Time_out;
var hexstring vc_Nid;
var hexstring vc_Nmk;
var hexstring vc_oui;
var hexstring vc_attenuation;
var AttenProfile_TYPE vc_attenuation_list;
var MACAddress_TYPE vc_sut_mac;
var MACAddress_TYPE vc_testSystem_plc_node_mac;
var MACAddressList_TYPE vc_macAddresList;
var float vc_LowestAverageAttenuation;
var RunID_TYPE vc_RunID;
var default vc_Default_SDP_Message;
var boolean vc_isTcpPortClosed;
var X509CertificateType vc_leafCert;
var X509CertificateType vc_subCert2;
var X509CertificateType vc_subCert1;
var idDigestValueMap vc_salesTariffIdDigestValueMap;
var idPrivateKeyMap vc_IdPrivateKeyMap;
var idCertificateMap vc_IdCertificateMap;
var float vc_processingTime;
var Sender_IPaddress_TYPE vc_sut_ipAddress;
var V2G_TCP_TLS_Parameter vc_v2g_tcp_tls_parameter;
var integer vc_maxRepetitionSDP := par_maxRepetitionSDP;
var IdType vc_SchemaID;
var GenChallengeType vc_GenChallenge;
var X509CertificateType vc_oEMProvisioningCert;
var XSDAUX.string vc_x509IssuerName;
var XSDAUX.integerXSD vc_x509SerialNumber;
var CertificateChainType vc_sAProvisioningCertChain;
var X509CertificateType vc_contractSignatureCert;
var CertificateChainType vc_contractSignatureCertChain;
var CertificateChainType vc_cpoEvseCertChain;
var X509CertificateType vc_v2gRootCert;
var X509CertificateType vc_peRootCert;
var ContractSignatureEncryptedPrivateKeyType vc_contractSignatureEncryptedPrivateKey;
var DiffieHellmanPublickeyType vc_dHpublickey;
var EMAIDType vc_eMAID;
var octetstring vc_sharedSecret;
var charstring vc_privateKey;
var Signature vc_signature;
var SelectedServiceListType vc_selectedServiceList;
var BodyType vc_myCertificateInstallationBodyType;
var BodyType vc_myCertificateUpdateBodyType;
var ChargingProfileType vc_chargingProfile;
var ServiceListType vc_serviceList;
var PaymentOptionListType vc_paymentOptionList;
var ChargeServiceType vc_chargeService;
var XSDAUX.unsignedShort vc_maxEntriesSAScheduleTuple;
var SupportedEnergyTransferModeType vc_supportedEnergyTransferMode;
var EnergyTransferModeType vc_requestedEnergyTransferMode;
var XSDAUX.unsignedInt vc_DepartureTime;
var PaymentOptionType vc_SelectedPaymentOption;
var SAScheduleListType vc_SASchedule_List;
var BodyType vc_myChargeParameterBodyType;
var PMaxScheduleType vc_PMaxSchedule;
var SalesTariffType vc_SalesTariff;
var ProfileEntryType vc_ProfileEntry_List;
var ParameterType vc_ParameterList;
var DC_EVSEStatusCodeType vc_EVSEStatusCode;

timer tc_TT_EVSE_SLAC_init;
timer tc_TP_EVSE_match_session;
timer tc_T_conn_resume;
timer tc_TT_EVSE_vald_toggle;
timer tc_T_step_EF;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
timer tc_SECC_B1_B2_Timer;
timer tc_TP_matching_repetition;
timer tc_TP_matching_rate;
timer tc_TP_EVSE_SLAC_init;
timer tc_SECC_waitForEIM;
timer tc_TconnResetup;
timer tc_T_step_X1;
timer tc_V2G_EVCC_Msg_Timer;
timer tc_V2G_EVCC_Ongoing_Timer;
timer tc_V2G_SECC_Ongoing_Timer;
timer tc_V2G_SECC_CableCheck_Timer;
timer tc_V2G_SECC_Msg_Timer;
timer tc_EVCC_DepartureTime_Timer;
timer tc_CMN_ProcessingTime_Timer;
timer tc_CMN_ScheduleListActive_Timer;

//initialization

var float vc_DutyCycle := 5.0;
var float vc_DutyCycleOscOff := 100.0;
var float vc_FrequencyOff := 0.0;
var float vc_DutyCycleE := 0.0;
var float vc_DutyCycleDelay := par_EVCC_setDC_delay;
var boolean vc_confirmState := false;
var boolean vc_confirmDC := false;
var HAL_61851_PwmMode_Type vc_errorState := e_OpenCircuit;
var boolean vc_testCaseSpecific := false;
var boolean vc_eimDone := false;
var boolean vc_isShutdownOSC := false;
var boolean vc_sleepAfterPlugOut := false;
var boolean vc_PtcIsValid := true;
var SourceRnd_Type vc_source_rnd := f_randomHexStringGen(32);
var boolean vc_unmapSlac := true;
var boolean vc_dutyCycleReceived := false;
var boolean vc_cpVoltageReceived := false;
var boolean vc_stateNotDetected := true;
var boolean vc_isProximityInvalid := false;
var boolean vc_powerUnitsInitialized := false;
var boolean vc_disconnectPowerUnits := false;
var boolean vc_linkDetected := false;
var hexstring par_cmValidate_result_notReady := '00'H;
var hexstring par_cmValidate_result_ready := '01'H;
var hexstring par_cmValidate_result_success := '02'H;
var hexstring par_cmValidate_result_failure := '03'H;
var hexstring par_cmValidate_result_notRequired := '04'H;
var boolean vc_activateNominal := true;
var boolean vc_TcpPortIsMapped := false;
var TransportProtocol_TYPE vc_TransportProtocol := '00'H;
var Security_TYPE vc_Security := '10'H;
var charstring vc_cipherSuites := par_cipherSuites;
var charstring vc_enabledProtocol := "TLSv1.2";
var integer vc_versionNumberMajor := 2;
var integer vc_versionNumberMinor := 0;
var SessionIDType vc_SessionID := '00'O;
var EvccIDType vc_eVCCID := hex2oct(par_testSystem_mac);
var EvseIDType vc_EvseId := "DE*BC4*E742*1234";
var EvseIDType cc_EvseId_NotSup := "ZZ00000";
var charstring vc_Supported_ISO_Namespace := "urn:iso:15118:2:2013:MsgDef";
var boolean vc_renegotiation := PICS_CMN_CMN_Renegotiation;
var boolean vc_sleepModeWoCharging := false;
var PhysicalValueType vc_remainingTimeToFullSoC := par_RemainingTimeToFullSoC;
var PhysicalValueType vc_remainingTimeToBulkSoC := par_RemainingTimeToBulkSoC;
var MeterInfoType vc_MeterInfo := { meterID := "Meter", meterReading := 0,
                                    sigMeterReading := omit,
                                       meterStatus := omit, tMeter := omit };
var boolean vc_receiptRequired := false;
var ServiceIDType vc_ServiceId := 1;
var boolean vc_isPowLimit := false;
var boolean vc_FreeService := true;
var EnergyTransferModeType vc_requestedEnergyTransferModeAC :=
                           par_requestedEnergyTransferModeAC;
var EnergyTransferModeType vc_requestedEnergyTransferModeDC :=
                           par_requestedEnergyTransferModeDC;
var PhysicalValueType vc_EVTargetCurrent := { multiplier := 0, unit := a, value_ := 1 };
var PhysicalValueType vc_EVTargetVoltage := par_EVMaximumVoltageLimit;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var PhysicalValueType vc_EAmount := { multiplier := 0, unit := wh, value_ := 100 };
var PhysicalValueType vc_EVMaxCurrent := par_EVMaxCurrent;
var PhysicalValueType vc_EVMinCurrent := par_EVMinCurrent;
var PhysicalValueType vc_EVMaxVoltage := par_EVMaxVoltage;
var PhysicalValueType vc_EVSEMaxCurrent := par_EVSEMaxCurrent;
var PhysicalValueType vc_EVSENominalVoltage := par_EVSENominalVoltage;
var PhysicalValueType vc_EVMaximumCurrentLimit := par_EVMaximumCurrentLimit;
var PhysicalValueType vc_EVMaximumVoltageLimit := par_EVMaximumVoltageLimit;
var PhysicalValueType vc_EVMaximumPowerLimit := par_EVMaximumPowerLimit;
var PhysicalValueType vc_EVSEPresentVoltage := { multiplier := 0, unit := v, value_ := 0 };
var PhysicalValueType vc_EVSEPresentCurrent := { multiplier := 0, unit := a, value_ := 0 };
var PhysicalValueType vc_EVSEMaximumCurrentLimit := par_EVSEMaximumCurrentLimit;
var PhysicalValueType vc_EVSEMaximumVoltageLimit := par_EVSEMaximumVoltageLimit;
var PhysicalValueType vc_EVSEMaximumPowerLimit := par_EVSEMaximumPowerLimit;
var PhysicalValueType vc_EVSEMinimumCurrentLimit := par_EVSEMinimumCurrentLimit;
var PhysicalValueType vc_EVSEMinimumVoltageLimit := par_EVSEMinimumVoltageLimit;
var PhysicalValueType vc_EVEnergyCapacity := { multiplier := 0, unit := wh, value_ := 10000 };
var PhysicalValueType vc_EVEnergyRequest := { multiplier := 0, unit := wh, value_ := 10000 };
var PercentValueType vc_fullSOC := 100;
var PercentValueType vc_bulkSOC := 80;
var PhysicalValueType vc_EVSECurrentRegulationTolerance := { multiplier := 0,
                                                              unit := a,
                                                              value_ := 1 };
var PhysicalValueType vc_EVSEPeakCurrentRipple := { multiplier := -1,
                                                    unit := a,
                                                    value_ := 15 };
var PhysicalValueType vc_EVSEEnergyToBeDelivered := { multiplier := 0,
                                                      unit := wh,
                                                      value_ := 10000 };
var PercentValueType  vc_EVRESSSOC := par_EVRESSSOC;
var SAIDType vc_SAScheduleTupleId := par_CMN_SAScheduleTuple;
var SAIDType vc_SAScheduleTupleId2 := vc_SAScheduleTupleId + 1;
var SAIDType vc_SAScheduleTupleId3 := vc_SAScheduleTupleId + 2;
var SAIDType vc_SAScheduleTupleIdRenegotiation := vc_SAScheduleTupleId + 3;
var SAIDType vc_SAScheduleTupleIdRenegotiation2 := vc_SAScheduleTupleId + 4;
var SAIDType vc_SAScheduleTupleIdRenegotiation3 := vc_SAScheduleTupleId + 5;
var SAIDType vc_SalesTariffId := 1;
var SAIDType vc_SalesTariffId2 := vc_SalesTariffId + 1;
var SAIDType vc_SalesTariffId3 := vc_SalesTariffId + 2;
var XSDAUX.short vc_ParameterSetId := 1;
var ChargingSessionType vc_ChargingSession  := terminate;
var float vc_scheduleListActiveTime := 0.0;
var charstring vc_contractCertIdName := "ccc_id";
var charstring vc_emaid_idName := "e_id";
var charstring vc_encryptedPrivateKeyIdName := "epk_id";
var charstring vc_dhPublicKeyIdName := "pk_id";
var charstring vc_salesTariffIdName := "ST";
var ChargeProgressType vc_ChargeProgress := start_;
var EVSENotificationType vc_EVSENotification := none_;
var AC_EVSEStatusType vc_AC_EVSEStatus := { rCD := false,
                                            notificationMaxDelay :=
                                            par_EVCC_NotificationMaxDelay,
                                            eVSENotification := none_ };
var DC_EVSEStatusType vc_DC_EVSEStatus := {notificationMaxDelay :=
                                           par_EVCC_NotificationMaxDelay,
                                           eVSENotification := none_,
                                           eVSEIsolationStatus := invalid,
                                           eVSEStatusCode := eVSE_Ready};
var DC_EVStatusType vc_DC_EVStatus := { eVReady:= false,
                                        eVErrorCode := nO_ERROR,
                                        eVRESSSOC := par_EVRESSSOC};
var IEC_61851_States vc_state := e_StateB;
var boolean vc_useSalesTariffs := PICS_EVCC_CMN_SalesTariff;

timer tc_V2G_EVCC_CommunicationSetup_Timer :=
      par_V2G_EVCC_CommunicationSetup_Timeout;
timer tc_V2G_SECC_CommunicationSetup_Timer :=
      par_V2G_SECC_CommunicationSetup_Performance_Time;
timer tc_TT_match_response := par_TT_match_response;
timer tc_TP_EV_batch_msg_interval := par_TP_EV_batch_msg_interval;
timer tc_TT_EVSE_match_MNBC := par_TT_EVSE_match_MNBC;
timer tc_TT_EV_atten_results := par_TT_EV_atten_results;
timer tc_TP_EVSE_atten_results := par_TT_EV_atten_results;
timer tc_TT_match_sequence := par_TT_match_sequence;
timer tc_TT_EVSE_match_session := par_TT_EVSE_match_session;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        timer tc_TT_match_join := par_TT_match_join;
        timer tc_TT_link_status_response := par_TT_link_status_response;
        timer tc_TP_match_response := par_TP_match_response;
        timer tc_TT_matching_repetition := par_TT_matching_repetition;
        timer tc_TT_link_status_polling_pause := par_TT_polling_pause;
        timer tc_T_match_leave := par_T_match_leave;
        timer tc_TP_EV_vald_toggle := par_TP_EV_vald_toggle;
        timer tc_TP_EV_vald_state_duration := par_TP_EV_vald_state_duration;
        timer tc_T_conn_max_comm := par_T_conn_max_comm;
        timer tc_TT_amp_map_exchange := par_TT_amp_map_exchange;
        timer tc_EVCC_StateB_Shutdown_Timer := par_EVCC_StateB_Shutdown_Timeout;
        timer tc_SECC_CPOscillator_Shutdown_Timer := par_SECC_CPOscillator_Shutdown_Timeout;
        timer tc_EVCC_SDP_Timer := par_EVCC_SDP_Timeout;
        timer tc_V2G_EVCC_PreCharge_Timer := par_V2G_EVCC_PreCharge_Timeout;
        timer tc_V2G_EVCC_CableCheck_Timer := par_V2G_EVCC_CableCheck_Timeout;
        timer tc_V2G_SECC_Sequence_Timeout := par_V2G_SECC_Sequence_Timeout;
        timer tc_V2G_SECC_Sequence_Timer := par_V2G_SECC_Sequence_Timeout;
        timer tc_CMN_TCP_Connection_Termination_Timer := par_CMN_TCP_Connection_Termination_Timeout;
    }
}
```

A.3 Timer configuration

```
module Timer_15118_2 {

    group Timer_par_15118_2 {

        //------------------ SECC sequence timeout ---------------
        modulepar float par_V2G_SECC_Sequence_Timeout := 60.0;

        //------------------ EVCC performance timeouts -----------------------
        modulepar float par_V2G_EVCC_Sequence_Performance_Time := 40.0;

        //------------------ SECC performance timeouts -----------------------
        modulepar float par_V2G_SECC_Msg_CurrentDemand_Performance_Time := 0.2;
        //in [s] Default value deviates from ISO 15118 norm
        modulepar float par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time := 1.5;

        //------------------ EVCC message timeouts ---------------------------
        modulepar float par_V2G_EVCC_Msg_Timeout_SupportedAppProtocolReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_SessionSetupReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_ServiceDetailReq := 5.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_CertificateInstallationReq := 5.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq := 5.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_PaymentDetailsReq := 5.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_AuthorizationReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq := 5.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_ChargingStatusReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_SessionStopReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_CableCheckReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_PreChargeReq := 2.0;
        modulepar float par_V2G_EVCC_Msg_Timeout_CurrentDemandReq := 0.25;
        modulepar float par_V2G_EVCC_Msg_Timeout_WeldingDetectionReq := 2.0;

        //------------------ EVCC Processing timeouts ------------------------
        modulepar float par_V2G_EVCC_Ongoing_Timeout := 60.0;
        modulepar float par_V2G_SECC_Ongoing_Performance_Timeout := 55.0;

        //------------------ Sequence CommunicationSetup timeouts ------------
        modulepar float par_V2G_EVCC_CommunicationSetup_Timeout := 20.0;
        modulepar float par_V2G_SECC_CommunicationSetup_Performance_Time := 18.0;

        //  ------------------ Sequence CableCheck timeouts --------------------
        modulepar float par_V2G_EVCC_CableCheck_Timeout := 40.0;

        //  ------------------ Sequence PreCharge timeouts ---------------------
        modulepar float par_V2G_EVCC_PreCharge_Timeout := 7.0;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //  ------------------ SDP timeout ------------------------------------
        modulepar float par_EVCC_SDP_Timeout := 0.25;

        //  ------------------ Non-standardised SECC Processing timeouts ------
        modulepar float par_SECC_chargingLoop_pause := 0.1;

        //  ------------------ Non-standardised EVCC Processing timeouts ------
        modulepar float par_V2G_SECC_CPState_Detection_Timeout := 0.25;
        modulepar float par_SECC_timeCableCheckVoltage := 3.0;
        modulepar float par_SECC_timeAuthenticationLoop := 3.0; //in [s]
        modulepar float par_SECC_CPOscillator_Shutdown_Timeout := 4.5;
        modulepar float par_EVCC_WeldingDetection_Timeout := 20.0; //in [s]
        modulepar float par_EVCC_StateB_Shutdown_Timeout := 4.5;
        modulepar float par_EVCC_StopCharging_Timeout := 2.0;

        //  ------------------ Non-standardised CMN Processing timeouts -------
        modulepar float par_CMN_TCP_Connection_Termination_Timeout := 5.0;

        modulepar float par_CMN_CMN_ChargingTime := 60.0;
    }
}



module Timer_15118 {

    group Timer_par_15118 {

        //  ------------------ Non-standardised CMN Processing timeouts -------
        modulepar float par_CMN_Transmission_Delay := 0.1;
        modulepar float par_CMN_HAL_Timeout := 2.0;
        modulepar float par_CMN_waitForNextHAL := 0.5;
        modulepar float par_CMN_Intern_PTC_Timeout := 0.4;
        modulepar float par_CMN_setKey := 3.0;
        modulepar float par_CMN_waitForPlugOut := 0.0; //in [s]
        modulepar float par_CMN_waitForSUTInitiatedWakeUp := 60.0;


        // ------------------ Non-standardised EVCC Processing timeouts -------
        modulepar float par_EVCC_waitForNextTC := 2.0;
        modulepar float par_EVCC_waitForPlugin := 60.0; //in [s]
        modulepar float par_EVCC_WakeUpOffsetTime := 10.0;

        // ------------------ Non-standardised SECC Processing timeouts -------
        modulepar float par_SECC_waitForNextTC := 20.0;
        modulepar float par_SECC_T_step_X1 := 3.0;

        modulepar float par_SECC_Pmax0W := 60.0;
    }
}
```

A.4 PICS configuration

```
module Pics_15118_2 {

    import from DataStructure_HAL_61851 all;
    import from DataStructure_PICS_15118_2 all;
    import from DataStructure_Security all;

    group PICS_15118_2 {


        group PICS_CMN {

            modulepar boolean PICS_CMN_CMN_Renegotiation := false;

            modulepar LoopCounter PICS_CMN_CMN_RenegotiationLoopIndication := 30;

            modulepar LoopCounter PICS_CMN_CMN_LoopCounter := 100;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        }
        group PICS_SECC_Tester {
            modulepar IEC_61851_States PICS_SECC_CMN_EvStateInChargeProgress := e_StateC;
            modulepar boolean PICS_SECC_DC_WeldingDetection := true;
            modulepar boolean PICS_SECC_CMN_DepartureTime := false;
            modulepar boolean PICS_SECC_CMN_MaxEntriesSAScheduleTuple := false;
            modulepar boolean PICS_SECC_CMN_OCSP := false;
        }
        group PICS_EVCC_Tester {
            modulepar boolean PICS_EVCC_CMN_ReceiptRequired := false;
            modulepar boolean PICS_EVCC_CMN_SalesTariff := false;

            modulepar boolean PICS_EVCC_CMN_OCSP := false;
        }
    }
}



module Pics_15118 {
    import from DataStructure_PICS_15118 all;
    group PICS_15118 {


        group PICS_CMN {
            modulepar boolean PICS_CMN_CMN_CombinedTesting := true;
            modulepar ChargingMode PICS_CMN_CMN_ChargingMode := aC;
            modulepar IdentificationMode PICS_CMN_CMN_IdentificationMode := eIM;
            modulepar PlugType PICS_CMN_CMN_PlugType := type2;
            modulepar CableCapabilityACType PICS_CMN_AC_CableCapability := capability32A;
            modulepar float PICS_CMN_CMN_WakeUp := 20.0;
        }
        group PICS_SECC_Tester {
            modulepar boolean PICS_SECC_CMN_Pause := false;
            modulepar EIMDone PICS_SECC_CMN_EIMDone := v2gAuthorization;
        }
        group PICS_EVCC_Tester {

            modulepar ZeroPow PICS_EVCC_CMN_PmaxSchedulewithZeroPow := none_;
        }
    }
}
```

A.5 PIXIT configuration

```
module Pixit_15118_2 {
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    import from DataStructure_PIXIT_15118_2 all;

    group PIXIT_15118_2 {

        group PIXIT_CMN {

            modulepar PkiType PIXIT_CMN_CMN_PkiName := PKI_1;

            modulepar Renegotiation PIXIT_CMN_CMN_Renegotiation := unknown;

            modulepar MeterReading PIXIT_CMN_CMN_MeterReading := unknown;
        }

        group PIXIT_SECC_Tester {

            modulepar VASSECC PIXIT_SECC_CMN_VAS := none_;

            modulepar ReceiptRequired PIXIT_SECC_CMN_ReceiptRequired := unknown;

            modulepar CertificateHandlingSECC PIXIT_SECC_CMN_CertificateHandling  := none_;

            modulepar SalesTariff PIXIT_SECC_CMN_SalesTariff := unknown;

            modulepar boolean PIXIT_SECC_CMN_TLS := false;

            modulepar integer PIXIT_SECC_CMN_VersionNumberMinor := 0;

            modulepar InternetAccess PIXIT_SECC_CMN_InternetAccess := unknown;

            modulepar Ongoing PIXIT_SECC_DC_CableCheckOngoing := unknown;

            modulepar Ongoing PIXIT_SECC_CMN_AuthorizationOngoing := unknown;

            modulepar Ongoing PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing := unknown;

            modulepar MeterInfo PIXIT_SECC_CMN_MeterInfo := unknown;

            modulepar PaymentOption PIXIT_SECC_CMN_PaymentOption := externalPayment;
        }

        group PIXIT_EVCC_Tester {

            modulepar VASEVCC PIXIT_EVCC_CMN_VAS := unknown;

            modulepar WeldingDetection PIXIT_EVCC_DC_WeldingDetection := unknown;

            modulepar TLS PIXIT_EVCC_CMN_TLS := unknown;

            modulepar CertificateHandlingEVCC PIXIT_EVCC_CMN_CertificateHandling := unknown;

            modulepar DepartureTime PIXIT_EVCC_CMN_DepartureTime := unknown;

            modulepar MaxNumberOfPhasesInUse PIXIT_EVCC_AC_MaxNumberOfPhasesInUse := unknown;

            modulepar ChargingProfileOpt PIXIT_EVCC_CMN_ChargingProfileOptimization := unknown;

            modulepar OldSAScheduleTupleID PIXIT_EVCC_CMN_OldSAScheduleTupleID := unknown;

            modulepar SAScheduleTupleIDMR PIXIT_EVCC_CMN_SAScheduleTupleIDMR := unknown;
        }
    }
}


module Pixit_15118 {

    import from DataStructure_PIXIT_15118 all;

    group PIXIT_15118 {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        group PIXIT_CMN {

            modulepar float PIXIT_CMN_CMN_WakeUp := 200.0;
        }

        group PIXIT_EVCC_Tester {


            modulepar Pause PIXIT_EVCC_CMN_Pause := unknown;
        }
    }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex B
(normative)

Test-case specifications for 15118-2 V2GTP

B.1 General information

This annex covers all V2GTP test case *specifications* in the ATS.

B.2 SECC test cases

```
module TestCases_SECC_CMN_CMN_V2GTPSessionSetup {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_V2GTPSessionSetup all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_V2GTPMessage all;

    testcase TC_SECC_CMN_VTB_V2GTPSessionSetup_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict == pass) {
            f_SECC_CMN_TB_VTB_V2GTPSessionSetup_001();
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        tc_V2G_EVCC_CommunicationSetup_Timer.stop;
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_SECC_CMN_VTB_V2GTPSessionSetup_002() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict == pass) {
             f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
                                                 '01'H,'FE'H,'8001'H, '00000000'H));
         }
         else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         tc_V2G_EVCC_CommunicationSetup_Timer.stop;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_V2GTPSessionSetup_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
                                                '01'H,'FE'H,'7001'H, omit));
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_V2GTPSessionSetup_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
                                                'FF'H,'FE'H,'8001'H, omit));
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_V2GTPSessionSetup_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
                                                '01'H,'FF'H,'8001'H, omit));
    }
    else {
        log("PreCondition was unsuccessful.");
    }


    //------------- Post Conditions-----------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     }
}



module TestCases_SECC_CMN_CMN_V2GTPSDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_V2GTPSDP all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_CMN_V2GTPMessage all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_SECC_CMN_VTB_V2GTPSDP_001() runs on SECC_Tester system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       //-------------- Pre Conditions-------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict == pass) {
               f_SECC_CMN_TB_VTB_V2GTPSDP_001();
       }
       else{
            log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_SECC_CMN_VTB_V2GTPSDP_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict == pass) {
             f_SECC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
                                            '01'H,'FE'H,'9000'H, '00000000'H));
        }
        else{
             log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_SECC_CMN_VTB_V2GTPSDP_003() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict == pass) {
             f_SECC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        '01'H,'FE'H,'7000'H, omit));
         }
         else{
              log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

      testcase TC_SECC_CMN_VTB_V2GTPSDP_004() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         //-------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict == pass) {
              f_SECC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
                                        'FF'H,'FE'H,'9000'H, omit));
         }
         else{
              log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

      testcase TC_SECC_CMN_VTB_V2GTPSDP_005() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         //-------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict == pass) {
              f_SECC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
                                        '01'H,'FF'H,'9000'H, omit));
         }
         else{
              log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}
```

B.3 EVCC test cases

```
module TestCases_EVCC_CMN_CMN_V2GTPSessionSetup {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_V2GTPSessionSetup all;
    import from Templates_CMN_V2GTPMessage all;

    testcase TC_EVCC_CMN_VTB_V2GTPSessionSetup_001() runs on EVCC_Tester system SystemEVCC {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_001();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_V2GTPSessionSetup_002() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
                                                 '01'H, 'FE'H, '8001'H, '00000000'H));
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_CMN_VTB_V2GTPSessionSetup_003() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
                                                 '01'H, 'FE'H, '7001'H, omit));
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_CMN_VTB_V2GTPSessionSetup_004() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
                                                    'FF'H, 'FE'H, '8001'H, omit));
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_V2GTPSessionSetup_005() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002(md_CMN_V2GTP_header_TYPE_001(
                                                    '01'H, 'FF'H, '8001'H, omit));
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_EVCC_CMN_CMN_V2GTPSDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_V2GTPSDP all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_V2GTPMessage all;

    testcase TC_EVCC_CMN_VTB_V2GTPSDP_001() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       //-------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict == pass) {
               f_EVCC_CMN_TB_VTB_V2GTPSDP_001();
       }
       else{
            log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_V2GTPSDP_002() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  var verdicttype preConVerdict;

  //-------------- Pre Conditions-------------------------------------------------------
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

  //-------------- Test behavior--------------------------------------------------------
  if(preConVerdict == pass) {
          f_EVCC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
                                          '01'H, 'FE'H, '9001'H, '00000000'H));
  }
  else{
       log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-----------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_V2GTPSDP_003() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   //-------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict == pass) {
       f_EVCC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
                                       '01'H, 'FE'H, '7001'H, omit));
   }
   else{
        log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-----------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_V2GTPSDP_004() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   //-------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict == pass) {
       f_EVCC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
                                       'FF'H, 'FE'H, '9001'H, omit));
   }
   else{
        log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-----------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_V2GTPSDP_005() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   //-------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict == pass) {
            f_EVCC_CMN_TB_VTB_V2GTPSDP_002(md_CMN_V2GTP_header_TYPE_001(
                                        '01'H, 'FF'H, '9001'H, omit));
        }
        else{
             log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex C
(normative)

Test-case specifications for 15118-2 SDP messages

C.1 General information

This annex covers all SDP test case *specifications* in the ATS.

C.2 SECC test cases

```
module TestCases_SECC_CMN_PnC_SDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_SDP all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

     testcase TC_SECC_CMN_VTB_SDP_003() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict == pass) {
             f_SECC_CMN_TB_VTB_SDP_001('00'H, fail);
        }
        else{
             log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_SECC_CMN_VTB_SDP_006() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict == pass) {
             f_SECC_CMN_TB_VTB_SDP_002('00'H);
        }
        else{
             log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}



module TestCases_SECC_CMN_EIM_SDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_SDP all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_SECC_CMN_VTB_SDP_001() runs on SECC_Tester system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       //-------------- Pre Conditions-------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict == pass) {
               f_SECC_CMN_TB_VTB_SDP_001('10'H, fail);
       }
       else{
            log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_SECC_CMN_VTB_SDP_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict == pass) {
             f_SECC_CMN_TB_VTB_SDP_001('00'H, fail);
        }
        else{
             log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_SECC_CMN_VTB_SDP_004() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict == pass) {
             f_SECC_CMN_TB_VTB_SDP_002('10'H);
        }
        else{
             log("PreCondition was unsuccessful.");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
         }

         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

      testcase TC_SECC_CMN_VTB_SDP_005() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         //-------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict == pass) {
              f_SECC_CMN_TB_VTB_SDP_002('00'H);
         }
         else{
              log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

      testcase TC_SECC_CMN_VTB_SDP_007() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         //-------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict == pass) {
              f_SECC_CMN_TB_VTB_SDP_003();
         }
         else{
              log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}
```

C.3 EVCC test cases

```
module TestCases_EVCC_DC_PnC_SDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_SDP all;
    import from PostConditions_EVCC_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from DataStructure_PICS_15118 all;

 testcase TC_EVCC_DC_VTB_SDP_004() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_checkWakeUpConfig(tsInitiated);
     if(preConVerdict == pass) {
        preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                         v_HAL_61851_Listener);
     }

     if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
     }

     if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
     }

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_SDP_002();
         if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                             "after previous SDP process.");
         }
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_DC_VTB_SDP_008() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_checkWakeUpConfig(sutInitiated);
     if(preConVerdict == pass) {
        preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                         v_HAL_61851_Listener);
     }

     if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
     }

     if(preConVerdict == pass) {
        preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
     }

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_SDP_002();
         if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                             "after previous SDP process.");
         }
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_SDP_012() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                            v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                            v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
               setverdict(pass, "The SUT has executed a reset of the message counter " &
                                "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_SDP_016() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                            v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                            v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
               setverdict(pass, "The SUT has executed a reset of the message counter " &
                                "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-----------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_EVCC_DC_EIM_SDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_SDP all;
    import from PostConditions_EVCC_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_EVCC_DC_VTB_SDP_001() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-----------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(tsInitiated);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                             v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
         }

         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_SDP_002();
             if(getverdict == pass) {
                setverdict(pass, "The SUT has executed a reset of the message counter " &
                                 "after previous SDP process.");
             }
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-----------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                        v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                        v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                        v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                        v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //------------- Post Conditions-------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                        v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior---------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                        v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_setPwmMode(e_PosVolt12);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
           v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
       } else {
           v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
       }
       preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                        v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                        v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
           v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
       } else {
           v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
       }
       preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                        v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SDP_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                        v_HAL_61851_Listener);
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                            v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
               setverdict(pass, "The SUT has executed a reset of the message counter " &
                                "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_SDP_013() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
        if(preConVerdict == pass) {
            preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                             v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                            v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
               setverdict(pass, "The SUT has executed a reset of the message counter " &
                                "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_SDP_014() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                            v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                            v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
               setverdict(pass, "The SUT has executed a reset of the message counter " &
                                "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_SDP_015() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                            v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                            v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
               setverdict(pass, "The SUT has executed a reset of the message counter " &
                                "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_EVCC_CMN_PnC_SDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_SDP all;
    import from PostConditions_EVCC_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;

     testcase TC_EVCC_CMN_VTB_SDP_004() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict == pass) {
             f_EVCC_CMN_TB_VTB_SDP_001('00'H, fail);
        }
        else{
             log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------------

        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}



module TestCases_EVCC_CMN_EIM_SDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_SDP all;
    import from PostConditions_EVCC_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from TestBehavior_EVCC_CommonBehavior all;
import from Timer_15118 all;
import from Pics_15118 all;
import from Pixit_15118 all;
import from Timer_15118_2 all;

testcase TC_EVCC_CMN_VTB_SDP_001() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   //-------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_SDP_001('10'H, fail);
   }
   else{
        log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------

   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_SDP_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //-------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
         f_EVCC_CMN_TB_VTB_SDP_001('00'H, fail);
    }
    else{
         log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------

    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_CMN_VTB_SDP_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //-------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
         f_EVCC_CMN_TB_VTB_SDP_001(?, fail);
    }
    else{
         log("PreCondition was unsuccessful.");
    }


    //------------- Post Conditions-------------------------------------------------------

    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}



module TestCases_EVCC_CMN_CMN_SDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_SDP all;
    import from PostConditions_EVCC_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;

    testcase TC_EVCC_CMN_VTB_SDP_005() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       //-------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict == pass) {
            f_EVCC_CMN_TB_VTB_SDP_002();
       }
       else{
            log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_SDP_006() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       //-------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict == pass) {
            f_EVCC_CMN_TB_VTB_SDP_003();
       }
       else{
            log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_SDP_007() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       //-------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict == pass) {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_EVCC_CMN_TB_VTB_SDP_004();
        }
        else{
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_EVCC_AC_PnC_SDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_SDP all;
    import from PostConditions_EVCC_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_EVCC_AC_VTB_SDP_004() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(tsInitiated);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
         }

         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_SDP_002();
             if(getverdict == pass) {
                setverdict(pass, "The SUT has executed a reset of the message counter " &
                                 "after previous SDP process.");
             }
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_EVCC_AC_VTB_SDP_008() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
           v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
       } else {
           v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
       }
       preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                        v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

    testcase TC_EVCC_AC_VTB_SDP_016() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
        if(preConVerdict == pass) {
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                            v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
               setverdict(pass, "The SUT has executed a reset of the message counter " &
                                "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_EVCC_AC_EIM_SDP {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_SDP all;
    import from PostConditions_EVCC_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_EVCC_AC_VTB_SDP_001() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
            setverdict(pass, "The SUT has executed a reset of the message counter " &
                             "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                            v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
               setverdict(pass, "The SUT has executed a reset of the message counter " &
                                "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-----------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_AC_VTB_SDP_010() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
        if(preConVerdict == pass) {
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                            v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
               setverdict(pass, "The SUT has executed a reset of the message counter " &
                                "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-----------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
           v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
       } else {
           v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
       }
       preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                        v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SDP_002();
        if(getverdict == pass) {
           setverdict(pass, "The SUT has executed a reset of the message counter " &
                            "after previous SDP process.");
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SDP_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
        preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
           v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
       } else {
           v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
       }
       preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                        v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
    }

    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
               setverdict(pass, "The SUT has executed a reset of the message counter " &
                                "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-----------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_AC_VTB_SDP_014() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
        if(preConVerdict == pass) {
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                            v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
               setverdict(pass, "The SUT has executed a reset of the message counter " &
                                "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-----------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_AC_VTB_SDP_015() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
        if(preConVerdict == pass) {
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            var float v_dataLinkprocessing;
            if(tc_CMN_ProcessingTime_Timer.running) {
                v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
            } else {
                v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
            }
            preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                             v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
            preConVerdict := f_EVCC_CMN_PR_PausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SDP_002();
            if(getverdict == pass) {
                setverdict(pass, "The SUT has executed a reset of the message counter " &
                                 "after previous SDP process.");
            }
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-----------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex D
(normative)

Test-case specifications for 15118-2 V2G application layer messages

D.1 General information

This annex covers all V2G application layer test case *specifications* in the ATS.

D.2 SECC test cases

D.2.1 V2G protocol handshake

D.2.1.1 SECC test cases for SupportedAppProtocol

```
module TestCases_SECC_CMN_EIM_SupportedAppProtocol {

    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_SupportedAppProtocol all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from urn_iso_15118_2_2010_AppProtocol all;
    import from Timer_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;

     testcase TC_SECC_CMN_VTB_SupportedAppProtocol_004() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_SECC_CMN_PR_InitConfiguration_002(system, pass, fail);
             if(getverdict == pass) {
                 f_SECC_CMN_TB_VTB_SupportedAppProtocol_001(fail);
                 tc_V2G_EVCC_CommunicationSetup_Timer.stop;
             }
          }
          else {
             log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}



module TestCases_SECC_CMN_CMN_SupportedAppProtocol {

    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_SupportedAppProtocol all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Configurations_15118_2 all;
import from urn_iso_15118_2_2010_AppProtocol all;
import from Timer_15118_2 all;
import from TestBehavior_SECC_CommonBehavior all;
import from LibFunctions_15118_2 all;
import from DataStructure_SDP all;
import from Templates_CMN_SDP all;

testcase TC_SECC_CMN_VTB_SupportedAppProtocol_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_PR_InitConfiguration_002(system, pass, fail);
        if(getverdict == pass) {
            f_SECC_CMN_TB_VTB_SupportedAppProtocol_001(fail);
            tc_V2G_EVCC_CommunicationSetup_Timer.stop;
        }
     }
     else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_SupportedAppProtocol_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_PR_InitConfiguration_002(system, pass, fail);
        if(getverdict == pass) {
            f_SECC_CMN_TB_VTB_SupportedAppProtocol_002();
            tc_V2G_EVCC_CommunicationSetup_Timer.stop;
        }
     }
     else {
        log("PreCondition was unsuccessful.");
     }

    //------------- Post Conditions-----------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_CMN_VTB_SupportedAppProtocol_005() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;
     var float v_processingTime := 0.5;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);

     alt {
       [] tc_V2G_SECC_CommunicationSetup_Timer.timeout {
          sleep(v_processingTime);
       }
       [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        ?, (49152 .. 65535), ?, ?)) {
                repeat;
              }
            }

            //-------------- Test behavior-------------------------------------------------------
            if(preConVerdict==pass){
               f_SECC_CMN_PR_InitConfiguration_002(system, fail, pass);
             }
             else {
                log("PreCondition was unsuccessful.");
            }

            //------------- Post Conditions------------------------------------------------------
            f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
            f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
        }

        testcase TC_SECC_CMN_VTB_SupportedAppProtocol_006() runs on SECC_Tester system SystemSECC {

            var HAL_61851_Listener v_HAL_61851_Listener;
            var verdicttype preConVerdict;

            // -------------- Pre Conditions------------------------------------------------------
            f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
            preConVerdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);

            //-------------- Test behavior-------------------------------------------------------
            if(preConVerdict==pass){
                f_SECC_CMN_PR_InitConfiguration_002(system, pass, fail);
                if(getverdict == pass) {
                    f_SECC_CMN_TB_VTB_SupportedAppProtocol_004();
                    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
                }
             }
             else {
                log("PreCondition was unsuccessful.");
            }


            //------------- Post Conditions------------------------------------------------------
            f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
            f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
        }
}
```

D.2.2 V2G messages

D.2.2.1 SECC test cases for SessionSetup

```
module TestCases_SECC_DC_CMN_SessionSetup {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_SessionSetup all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Timer_15118_2 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_SECC_DC_VTB_SessionSetup_001() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
       preConVerdict :=
       f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_005(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionSetup_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
       preConVerdict :=
       f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(false);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionSetup_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
       preConVerdict :=
       f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(true);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionSetup_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
       preConVerdict :=
       f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_005(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionSetup_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiated);
        if(preConVerdict == pass) {
           preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
           preConVerdict :=
           f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict == pass) {
            f_SECC_CMN_TB_VTB_SessionSetup_006(false);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        tc_V2G_EVCC_CommunicationSetup_Timer.stop;
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_DC_VTB_SessionSetup_006() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiated);
        if(preConVerdict == pass) {
           preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
           preConVerdict :=
           f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict == pass) {
            f_SECC_CMN_TB_VTB_SessionSetup_006(true);
        }
        else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        tc_V2G_EVCC_CommunicationSetup_Timer.stop;
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_SECC_CMN_CMN_SessionSetup {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from DataStructure_HAL_61851 all;
import from TestBehavior_SECC_SessionSetup all;
import from Services_HAL_61851_1 all;
import from TestBehavior_SECC_CommonBehavior all;
import from PreConditions_SECC_15118_2 all;
import from PostConditions_SECC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;
import from LibFunctions_15118_2 all;
import from Timer_15118_2 all;
import from TestBehavior_SECC_CommonBehavior all;
import from Timer_15118 all;
import from Pics_15118 all;
import from Pixit_15118 all;

testcase TC_SECC_CMN_VTB_SessionSetup_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);


    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_001(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_SessionSetup_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);


    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_002();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_CMN_VTB_SessionSetup_004() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);


     //-------------- Test behavior--------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(preConVerdict == pass) {
            f_SECC_CMN_TB_VTB_SessionSetup_004();
        }
        else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_SECC_AC_CMN_SessionSetup {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_SessionSetup all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Timer_15118_2 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_SECC_AC_VTB_SessionSetup_001() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(tsInitiated);
         if(preConVerdict == pass) {
            preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
             preConVerdict :=
             f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior---------------------------------------------------------------
         if(preConVerdict == pass) {
             f_SECC_CMN_TB_VTB_SessionSetup_005(fail);
         }
         else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------------
         tc_V2G_EVCC_CommunicationSetup_Timer.stop;
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_SECC_AC_VTB_SessionSetup_002() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
       preConVerdict :=
       f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(false);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionSetup_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
       preConVerdict :=
       f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(true);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionSetup_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
       preConVerdict :=
       f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_005(fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionSetup_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
       preConVerdict :=
       f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict == pass) {
        f_SECC_CMN_TB_VTB_SessionSetup_006(false);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    tc_V2G_EVCC_CommunicationSetup_Timer.stop;
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_SessionSetup_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiated);
        if(preConVerdict == pass) {
           preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
           preConVerdict :=
           f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict == pass) {
            f_SECC_CMN_TB_VTB_SessionSetup_006(true);
        }
        else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions----------------------------------------------------
        tc_V2G_EVCC_CommunicationSetup_Timer.stop;
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```

D.2.2.2 SECC test cases for ServiceDiscovery

```
module TestCases_SECC_DC_CMN_ServiceDiscovery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_ServiceDiscovery all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from Timer_15118_2 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_SECC_DC_VTB_ServiceDiscovery_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(tsInitiated);
        if(preConVerdict == pass) {
           preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
        }

        if(preConVerdict == pass) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
            preConVerdict := f_SECC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_ServiceDiscovery_005(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_DC_VTB_ServiceDiscovery_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiated);
        if(preConVerdict == pass) {
           preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
           preConVerdict := f_SECC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_ServiceDiscovery_005(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_SECC_CMN_CMN_ServiceDiscovery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_ServiceDiscovery all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;

    testcase TC_SECC_CMN_VTB_ServiceDiscovery_001() runs on SECC_Tester system SystemSECC {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDiscovery_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDiscovery_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDiscovery_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDiscovery_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_ServiceDiscovery_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }


        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_SECC_AC_CMN_ServiceDiscovery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_ServiceDiscovery all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from Timer_15118_2 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_SECC_AC_VTB_ServiceDiscovery_001() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(tsInitiated);
         if(preConVerdict == pass) {
            preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
            preConVerdict := f_SECC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior---------------------------------------------------------
         if(preConVerdict==pass){
             f_SECC_CMN_TB_VTB_ServiceDiscovery_005(fail);
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_SECC_AC_VTB_ServiceDiscovery_002() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(sutInitiated);
         if(preConVerdict == pass) {
            preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
         }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
           preConVerdict := f_SECC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_ServiceDiscovery_005(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-----------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```

D.2.2.3 SECC test cases for ServiceDetail

```
module TestCases_SECC_CMN_PnC_ServiceDetail {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_ServiceDetail all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_SECC_CMN_VTB_ServiceDetail_002() runs on SECC_Tester system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-----------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

       //-------------- Test behavior-------------------------------------------------------
       if(preConVerdict==pass){
           f_SECC_CMN_TB_VTB_ServiceDetail_001(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-----------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

   testcase TC_SECC_CMN_VTB_ServiceDetail_004() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-----------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

      //-------------- Test behavior-------------------------------------------------------
      if(preConVerdict==pass){
          f_SECC_CMN_TB_VTB_ServiceDetail_002();
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      }else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

  testcase TC_SECC_CMN_VTB_ServiceDetail_006() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_SECC_CMN_TB_VTB_ServiceDetail_003();
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

  testcase TC_SECC_CMN_VTB_ServiceDetail_008() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_SECC_CMN_TB_VTB_ServiceDetail_004();
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

  testcase TC_SECC_CMN_VTB_ServiceDetail_010() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_SECC_CMN_TB_VTB_ServiceDetail_005();
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

   testcase TC_SECC_CMN_VTB_ServiceDetail_012() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
          f_SECC_CMN_TB_VTB_ServiceDetail_001(fail);
      }else {
          log("PreCondition was unsuccessful.");
      }


      //------------- Post Conditions------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }
}



module TestCases_SECC_CMN_EIM_ServiceDetail {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_ServiceDetail all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_SECC_CMN_VTB_ServiceDetail_001() runs on SECC_Tester system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict==pass){
           f_SECC_CMN_TB_VTB_ServiceDetail_001(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions------------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

   testcase TC_SECC_CMN_VTB_ServiceDetail_003() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
          f_SECC_CMN_TB_VTB_ServiceDetail_002();
      }else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 testcase TC_SECC_CMN_VTB_ServiceDetail_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_CMN_VTB_ServiceDetail_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_CMN_VTB_ServiceDetail_009() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_ServiceDetail_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

   testcase TC_SECC_CMN_VTB_ServiceDetail_011() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_SECC_CMN_TB_VTB_ServiceDetail_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```

D.2.2.4 SECC test cases for PaymentServiceSelection

```
module TestCases_SECC_CMN_PnC_PaymentServiceSelection {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_PaymentServiceSelection all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

     testcase TC_SECC_CMN_VTB_PaymentServiceSelection_003() runs on SECC_Tester
                                                             system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-----------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001 (v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

        //-------------- Test behavior-----------------------------------------------------
        if ( preConVerdict == pass ) {
            f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(fail);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-----------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_SECC_CMN_VTB_PaymentServiceSelection_010() runs on SECC_Tester
                                                            system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions----------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

       //-------------- Test behavior----------------------------------------------------
       if ( preConVerdict == pass ) {
           f_SECC_CMN_TB_VTB_PaymentServiceSelection_007();
       }
       else {
           log ("PreCondition was unsuccessful.");
       }


       //------------- Post Conditions----------------------------------------------------
       f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }
 }



module TestCases_SECC_CMN_EIM_PaymentServiceSelection {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_PaymentServiceSelection all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

     testcase TC_SECC_CMN_VTB_PaymentServiceSelection_002() runs on SECC_Tester
                                                             system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-----------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001 (v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

        //-------------- Test behavior-----------------------------------------------------
        if ( preConVerdict == pass ) {
            f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(fail);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-----------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_SECC_CMN_VTB_PaymentServiceSelection_009() runs on SECC_Tester
                                                            system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions----------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

       //-------------- Test behavior-----------------------------------------------------
       if ( preConVerdict == pass ) {
           f_SECC_CMN_TB_VTB_PaymentServiceSelection_007();
       }
       else {
           log ("PreCondition was unsuccessful.");
       }


       //------------- Post Conditions-----------------------------------------------------
       f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
 }



module TestCases_SECC_CMN_CMN_PaymentServiceSelection {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_PaymentServiceSelection all;
    import from Services_HAL_61851_1 all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from TestBehavior_SECC_CommonBehavior all;
import from PreConditions_SECC_15118_2 all;
import from PostConditions_SECC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;

  testcase TC_SECC_CMN_VTB_PaymentServiceSelection_001() runs on SECC_Tester
                                                            system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001 (v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------
     if ( preConVerdict == pass ) {
         f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(fail);
     }
     else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

  testcase TC_SECC_CMN_VTB_PaymentServiceSelection_004() runs on SECC_Tester
                                                            system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------
   if ( preConVerdict == pass ) {
       f_SECC_CMN_TB_VTB_PaymentServiceSelection_002();
   }
   else {
       log ("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------
   f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_CMN_VTB_PaymentServiceSelection_005() runs on SECC_Tester
                                                           system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------
   if ( preConVerdict == pass ) {
       f_SECC_CMN_TB_VTB_PaymentServiceSelection_003();
   }
   else {
       log ("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------
   f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  testcase TC_SECC_CMN_VTB_PaymentServiceSelection_006() runs on SECC_Tester
                                                          system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------
    if ( preConVerdict == pass ) {
        f_SECC_CMN_TB_VTB_PaymentServiceSelection_004();
    }
    else {
        log ("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------
    f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

  testcase TC_SECC_CMN_VTB_PaymentServiceSelection_007() runs on SECC_Tester
                                                          system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------
    if ( preConVerdict == pass ) {
        f_SECC_CMN_TB_VTB_PaymentServiceSelection_005();
    }
    else {
        log ("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------
    f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

  testcase TC_SECC_CMN_VTB_PaymentServiceSelection_008() runs on SECC_Tester
                                                          system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001( v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------
    if ( preConVerdict == pass ) {
        f_SECC_CMN_TB_VTB_PaymentServiceSelection_006();
    }
    else {
        log ("PreCondition was unsuccessful.");
    }


    //------------- Post Conditions-------------------------------------------------
    f_SECC_CMN_PO_InitialState_001( v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

D.2.2.5 SECC test cases for PaymentDetails

```
module TestCases_SECC_CMN_PnC_PaymentDetails {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_PaymentDetails all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_Security all;

   testcase TC_SECC_CMN_VTB_PaymentDetails_001() runs on SECC_Tester system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict==pass){
           f_SECC_CMN_TB_VTB_PaymentDetails_001(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

   testcase TC_SECC_CMN_VTB_PaymentDetails_002() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
          f_SECC_CMN_TB_VTB_PaymentDetails_001(fail);
      }else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

   testcase TC_SECC_CMN_VTB_PaymentDetails_003() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_SECC_CMN_TB_VTB_PaymentDetails_001(fail);
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

  testcase TC_SECC_CMN_VTB_PaymentDetails_004() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
          f_SECC_CMN_TB_VTB_PaymentDetails_002();
      }else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

    testcase TC_SECC_CMN_VTB_PaymentDetails_005() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_PaymentDetails_003();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_SECC_CMN_VTB_PaymentDetails_006() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_SECC_CMN_TB_VTB_PaymentDetails_004();
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

    testcase TC_SECC_CMN_VTB_PaymentDetails_007() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_PaymentDetails_005(cc_crt_contract_leaf_expired,
                                             cc_crt_mo_sub2_valid,
                                             cc_crt_mo_sub1_valid,
                                             fAILED_CertificateExpired);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentDetails_008() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_PaymentDetails_005(cc_crt_contract_leaf_invalid_content,
                                             cc_crt_mo_sub2_valid,
                                             cc_crt_mo_sub1_valid,
                                             m_CMN_failedResponseCodeList_003());
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentDetails_009() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_PaymentDetails_005(
                            cc_crt_contract_leaf_expired_issuer_sub1,
                            cc_crt_mo_sub2_expired,
                            cc_crt_mo_sub1_valid,
                            m_CMN_failedResponseCodeList_003());
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_PaymentDetails_010() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_SECC_CMN_TB_VTB_PaymentDetails_005(
                                cc_crt_contract_leaf_expired_issuer_sub2,
                                cc_crt_mo_sub2_valid,
                                cc_crt_mo_sub1_expired,
                                m_CMN_failedResponseCodeList_003());
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```

D.2.2.6 SECC test cases for Authorization

```
module TestCases_SECC_CMN_PnC_Authorization {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_Authorization all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_SECC_CMN_VTB_Authorization_002() runs on SECC_Tester system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------------
       if(preConVerdict==pass){
           f_SECC_CMN_TB_VTB_Authorization_002(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

  testcase TC_SECC_CMN_VTB_Authorization_004() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------------
     if(preConVerdict==pass){
         f_SECC_CMN_TB_VTB_Authorization_003();
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 testcase TC_SECC_CMN_VTB_Authorization_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_CMN_VTB_Authorization_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_006();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_CMN_VTB_Authorization_009() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

   testcase TC_SECC_CMN_VTB_Authorization_010() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_SECC_CMN_TB_VTB_Authorization_008();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_CMN_VTB_Authorization_011() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_009();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_CMN_VTB_Authorization_012() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_002(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_CMN_VTB_Authorization_013() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_002(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_CMN_VTB_Authorization_015() runs on SECC_Tester system SystemSECC {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_Authorization_011();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_Authorization_016() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_Authorization_011();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_CMN_VTB_Authorization_017() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_Authorization_011();
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_SECC_CMN_EIM_Authorization {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_Authorization all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   import from ComponentsAndPorts all;
   import from Configurations_15118_2 all;

   testcase TC_SECC_CMN_VTB_Authorization_001() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
          f_SECC_CMN_TB_VTB_Authorization_001(fail);
      }else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

  testcase TC_SECC_CMN_VTB_Authorization_003() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_SECC_CMN_TB_VTB_Authorization_003();
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_CMN_VTB_Authorization_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_Authorization_008() runs on SECC_Tester system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_Authorization_007();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

   testcase TC_SECC_CMN_VTB_Authorization_014() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

      //-------------- Test behavior-------------------------------------------------------------
      if(preConVerdict==pass){
          f_SECC_CMN_TB_VTB_Authorization_010();
      }else {
          log("PreCondition was unsuccessful.");
      }


      //------------- Post Conditions------------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }
}
```

D.2.2.7 SECC test cases for ChargeParameterDiscovery

```
module TestCases_SECC_DC_CMN_ChargeParameterDiscovery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_ChargeParameterDiscovery all;
    import from ComponentsAndPorts all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from Configurations_15118_2 all;
    import from Timer_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_001() runs on SECC_Tester
                                                              system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

       //-------------- Test behavior-------------------------------------------------------------
       if(preConVerdict==pass){
           f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions------------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_003() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_004() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_005() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
        f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_006() runs on SECC_Tester
                                                          system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_007() runs on SECC_Tester
                                                          system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
         preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_008() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_009() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_010() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_011() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
      preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_012() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_013() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_014() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_008();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_015() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_016() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_017() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
   }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior-------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_018() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(tsInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior-------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_019() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(tsInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
  }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior-------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions----------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_020() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-----------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(tsInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior-------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions----------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_021() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-----------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(tsInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  //-------------- Test behavior-------------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_022() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions------------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(tsInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior-------------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_023() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_024() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_025() runs on SECC_Tester
                                                       system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_026() runs on SECC_Tester
                                                       system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_027() runs on SECC_Tester
                                                       system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   if(preConVerdict==pass){
       vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_028() runs on SECC_Tester
                                                       system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 22 };
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_029() runs on SECC_Tester
                                                       system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(sutInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior--------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_030() runs on SECC_Tester
                                                       system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(sutInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior--------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_031() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(sutInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior--------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_032() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(sutInitiated);
  if(preConVerdict == pass) {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
      preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_033() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
      preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_034() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
      preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions----------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_035() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-----------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
      preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions----------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_ChargeParameterDiscovery_036() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-----------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_DC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005();
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-----------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}



module TestCases_SECC_CMN_CMN_ChargeParameterDiscovery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_ChargeParameterDiscovery all;
    import from ComponentsAndPorts all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from Configurations_15118_2 all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;

     testcase TC_SECC_CMN_VTB_ChargeParameterDiscovery_001() runs on SECC_Tester
                                                              system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-----------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_001();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-----------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_SECC_CMN_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
                                                              system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-----------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_002();
        }else {
            log("PreCondition was unsuccessful.");
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //------------- Post Conditions-------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_SECC_AC_CMN_ChargeParameterDiscovery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_ChargeParameterDiscovery all;
    import from ComponentsAndPorts all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from Configurations_15118_2 all;
    import from Timer_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_001() runs on SECC_Tester
                                                             system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------------
       if(preConVerdict==pass){
           f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
                                                            system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------------
       if(preConVerdict==pass){
           vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
           f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

     testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_003() runs on SECC_Tester
                                                             system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_002();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_004() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_005() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_006() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
   }else {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

  testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_007() runs on SECC_Tester
                                                          system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
          preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

  testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_008() runs on SECC_Tester
                                                          system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

  testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_009() runs on SECC_Tester
                                                          system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_010() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_011() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
      preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_012() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_013() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_014() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_008();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_015() runs on SECC_Tester
                                                         system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_016() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_017() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
       preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_018() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(tsInitiated);
    if(preConVerdict == pass) {
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
      preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_019() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
      preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_020() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
      preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_021() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
      preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_022() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_SECC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior-------------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_023() runs on SECC_Tester
                                                       system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_024() runs on SECC_Tester
                                                       system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_025() runs on SECC_Tester
                                                       system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
       f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_026() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_027() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_028() runs on SECC_Tester
                                                         system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 32 };
        f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_029() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(sutInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior--------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_030() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(sutInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior--------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_031() runs on SECC_Tester
                                                        system SystemSECC {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(sutInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior--------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_032() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(sutInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior--------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_033() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(sutInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior-------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_034() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(sutInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
  }

  if(preConVerdict == pass) {
     map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
     preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
  }

  if(preConVerdict == pass) {
     unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
     preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
  }

  //-------------- Test behavior-------------------------------------------------------
  if(preConVerdict==pass){
      f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_035() runs on SECC_Tester
                                                        system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_checkWakeUpConfig(sutInitiated);
  if(preConVerdict == pass) {
     preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
  }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
           preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

      testcase TC_SECC_AC_VTB_ChargeParameterDiscovery_036() runs on SECC_Tester
                                                              system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiated);
        if(preConVerdict == pass) {
           preConVerdict := f_SECC_AC_PR_SessionStop_001(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_SECC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
           preConVerdict := f_SECC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005();
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}
```

D.2.2.8 SECC test cases for PowerDelivery

```
module TestCases_SECC_DC_CMN_PowerDelivery {

    import from DataStructure_HAL_61851 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from Pics_15118_2 all;
    import from TestBehavior_SECC_PowerDelivery all;
    import from ComponentsAndPorts all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  import from Configurations_15118_2 all;

 testcase TC_SECC_DC_VTB_PowerDelivery_001() runs on SECC_Tester system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

  //-------------- Test behavior--------------------------------------------------------
  if ( preConVerdict == pass ){
      f_SECC_DC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
                                         false, true, fail);
  }
  else {
    log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
 }

 testcase TC_SECC_DC_VTB_PowerDelivery_002() runs on SECC_Tester system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict :=
  f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

  //-------------- Test behavior--------------------------------------------------------
  if ( preConVerdict == pass ){
    vc_DC_EVStatus.eVReady := false;
    f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                       true, true, fail);
  }
  else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
 }

testcase TC_SECC_DC_VTB_PowerDelivery_003() runs on SECC_Tester system SystemSECC {

 var HAL_61851_Listener v_HAL_61851_Listener;
 var verdicttype preConVerdict;

 // -------------- Pre Conditions-------------------------------------------------------
 f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
 preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

 //-------------- Test behavior--------------------------------------------------------
 if ( preConVerdict == pass ){
     f_SECC_DC_TB_VTB_PowerDelivery_002();
 }
 else {
   log("PreCondition was unsuccessful.");
 }

 //------------- Post Conditions-------------------------------------------------------
 f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
 f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

 testcase TC_SECC_DC_VTB_PowerDelivery_004() runs on SECC_Tester system SystemSECC {

 var HAL_61851_Listener v_HAL_61851_Listener;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict :=
  f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

  //-------------- Test behavior--------------------------------------------------------
  if ( preConVerdict == pass ){
    vc_DC_EVStatus.eVReady := false;
    f_SECC_DC_TB_VTB_PowerDelivery_002();
  }
  else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-----------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
 }

   testcase TC_SECC_DC_VTB_PowerDelivery_005() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ){
         f_SECC_DC_TB_VTB_PowerDelivery_003(start_);
     }
     else {
       log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-----------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
   }

testcase TC_SECC_DC_VTB_PowerDelivery_006()  runs on SECC_Tester system SystemSECC {

 var HAL_61851_Listener v_HAL_61851_Listener;
 var verdicttype preConVerdict;

 // -------------- Pre Conditions-------------------------------------------------------
 f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
 preConVerdict :=
 f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

 //-------------- Test behavior--------------------------------------------------------
 if ( preConVerdict == pass ){
   f_SECC_DC_TB_VTB_PowerDelivery_003(stop_);
 }
 else {
     log("PreCondition was unsuccessful.");
 }

 //------------- Post Conditions-----------------------------------------------------
 f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
 f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

   testcase TC_SECC_DC_VTB_PowerDelivery_007() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  //-------------- Test behavior-------------------------------------------------------
  if ( preConVerdict == pass ){
      f_SECC_DC_TB_VTB_PowerDelivery_004(start_);
  }
  else {
    log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_008() runs on SECC_Tester system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

  //-------------- Test behavior-------------------------------------------------------
  if ( preConVerdict == pass ){
      vc_DC_EVStatus.eVReady := true;
      f_SECC_DC_TB_VTB_PowerDelivery_005(start_);
  }
  else {
    log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_009() runs on SECC_Tester system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

  //-------------- Test behavior-------------------------------------------------------
  if ( preConVerdict == pass ){
      f_SECC_CMN_TB_VTB_PowerDelivery_001();
  }
  else {
    log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

 testcase TC_SECC_DC_VTB_PowerDelivery_010() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict :=
      f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

      //-------------- Test behavior-------------------------------------------------------
      if ( preConVerdict == pass ){
        f_SECC_CMN_TB_VTB_PowerDelivery_001();
      }
      else {
          log("PreCondition was unsuccessful.");
      }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      //------------- Post Conditions-------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
 }

 testcase TC_SECC_DC_VTB_PowerDelivery_011() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_DC_PR_CurrentDemandOrMeteringReceiptRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_PowerDelivery_001(renegotiate, v_HAL_61851_Listener,
                                           false, true, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

 testcase TC_SECC_DC_VTB_PowerDelivery_012() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict :=
     f_SECC_DC_PR_ChargeParameterDiscoveryRenegotiation_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_SECC_DC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
                                            true, true, fail);
     }
     else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
  }

 testcase TC_SECC_DC_VTB_PowerDelivery_013() runs on SECC_Tester system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if ( preConVerdict == pass ){
       f_SECC_DC_TB_VTB_PowerDelivery_006();
   }
   else {
     log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
 }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
testcase TC_SECC_DC_VTB_PowerDelivery_014() runs on SECC_Tester system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict :=
  f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

  //-------------- Test behavior--------------------------------------------------------
  if ( preConVerdict == pass ){
    vc_DC_EVStatus.eVReady := false;
    vc_DC_EVStatus.eVErrorCode := fAILED_RESSTemperatureInhibit;
    f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                       true, true, fail);
  }
  else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-----------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_015() runs on SECC_Tester system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict :=
  f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

  //-------------- Test behavior--------------------------------------------------------
  if ( preConVerdict == pass ){
    vc_DC_EVStatus.eVReady := false;
    vc_DC_EVStatus.eVErrorCode := fAILED_EVShiftPosition;
    f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                       true, true, fail);
  }
  else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-----------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_016() runs on SECC_Tester system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict :=
  f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

  //-------------- Test behavior--------------------------------------------------------
  if ( preConVerdict == pass ){
    vc_DC_EVStatus.eVReady := false;
    vc_DC_EVStatus.eVErrorCode := fAILED_ChargerConnectorLockFault;
    f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                       true, true, fail);
  }
  else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-----------------------------------------------------
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_017() runs on SECC_Tester system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict :=
  f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

  //-------------- Test behavior--------------------------------------------------------
  if ( preConVerdict == pass ){
    vc_DC_EVStatus.eVReady := false;
    vc_DC_EVStatus.eVErrorCode := fAILED_EVRESSMalfunction;
    f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                       true, true, fail);
  }
  else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_018() runs on SECC_Tester system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict :=
  f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

  //-------------- Test behavior--------------------------------------------------------
  if ( preConVerdict == pass ){
    vc_DC_EVStatus.eVReady := false;
    vc_DC_EVStatus.eVErrorCode := fAILED_ChargingCurrentdifferential;
    f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                       true, true, fail);
  }
  else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_PowerDelivery_019() runs on SECC_Tester system SystemSECC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict :=
  f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

  //-------------- Test behavior--------------------------------------------------------
  if ( preConVerdict == pass ){
    vc_DC_EVStatus.eVReady := false;
    vc_DC_EVStatus.eVErrorCode := fAILED_ChargingVoltageOutOfRange;
    f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                       true, true, fail);
  }
  else {
      log("PreCondition was unsuccessful.");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
      }

      testcase TC_SECC_DC_VTB_PowerDelivery_020() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if ( preConVerdict == pass ){
          vc_DC_EVStatus.eVReady := false;
          vc_DC_EVStatus.eVErrorCode := fAILED_ChargingSystemIncompatibility;
          f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                             true, true, fail);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
      }

      testcase TC_SECC_DC_VTB_PowerDelivery_021() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if ( preConVerdict == pass ){
          vc_DC_EVStatus.eVReady := false;
          vc_DC_EVStatus.eVErrorCode := noData;
          f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                             true, true, fail);
        }
        else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
      }
}



module TestCases_SECC_AC_CMN_PowerDelivery {

    import from DataStructure_HAL_61851 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from Pics_15118_2 all;
    import from TestBehavior_SECC_PowerDelivery all;
    import from ComponentsAndPorts all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from Configurations_15118_2 all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 testcase TC_SECC_AC_VTB_PowerDelivery_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
  }

 testcase TC_SECC_AC_VTB_PowerDelivery_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
 }

 testcase TC_SECC_AC_VTB_PowerDelivery_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_AC_PR_ChargingStatusOrMeteringReceiptRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_001(renegotiate, v_HAL_61851_Listener, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_SECC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener, fail);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
  }

  testcase TC_SECC_AC_VTB_PowerDelivery_005() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict :=
     f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(v_HAL_61851_Listener);

     //-------------- Test behavior-------------------------------------------------------
     if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener, fail);
     }
     else {
       log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
 }

 testcase TC_SECC_AC_VTB_PowerDelivery_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_002(start_, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_AC_VTB_PowerDelivery_007() runs on SECC_Tester system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict :=
   f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_AC_TB_VTB_PowerDelivery_002(stop_, v_HAL_61851_Listener);
   }
   else {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

     testcase TC_SECC_AC_VTB_PowerDelivery_008() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_AC_TB_VTB_PowerDelivery_002(start_, v_HAL_61851_Listener);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

    testcase TC_SECC_AC_VTB_PowerDelivery_009() runs on SECC_Tester system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict :=
       f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict==pass){
           f_SECC_AC_TB_VTB_PowerDelivery_003(stop_, v_HAL_61851_Listener);
       }
       else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

     testcase TC_SECC_AC_VTB_PowerDelivery_010() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_AC_TB_VTB_PowerDelivery_004(start_, v_HAL_61851_Listener);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 testcase TC_SECC_AC_VTB_PowerDelivery_011() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_005(start_, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

 testcase TC_SECC_AC_VTB_PowerDelivery_012() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_PowerDelivery_001();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

 testcase TC_SECC_AC_VTB_PowerDelivery_013() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_PowerDelivery_001();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

 testcase TC_SECC_AC_VTB_PowerDelivery_014() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_PowerDelivery_006(renegotiate, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }


    //------------- Post Conditions------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
  }
}
```

D.2.2.9 SECC test cases for CertificateUpdate

```
module TestCases_SECC_CMN_PnC_CertificateUpdate {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_CertificateUpdate all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_Security all;

   testcase TC_SECC_CMN_VTB_CertificateUpdate_001() runs on SECC_Tester system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions------------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

       //-------------- Test behavior-------------------------------------------------------------
       if(preConVerdict==pass){
           f_SECC_CMN_TB_VTB_CertificateUpdate_001(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions------------------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

   testcase TC_SECC_CMN_VTB_CertificateUpdate_002() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions------------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

      //-------------- Test behavior-------------------------------------------------------------
      if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateUpdate_002();
      }else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions------------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 testcase TC_SECC_CMN_VTB_CertificateUpdate_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_004() runs on SECC_Tester system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
      f_SECC_CMN_TB_VTB_CertificateUpdate_004();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

  testcase TC_SECC_CMN_VTB_CertificateUpdate_005() runs on SECC_Tester
                                                    system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_SECC_CMN_TB_VTB_CertificateUpdate_005();
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

    testcase TC_SECC_CMN_VTB_CertificateUpdate_006() runs on SECC_Tester
                                                      system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_006();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_008() runs on SECC_Tester
                                                 system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_007("WrongPKILeafCert");
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_009() runs on SECC_Tester
                                                 system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_007("WrongPKISub2Cert");
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_010() runs on SECC_Tester
                                                 system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_007("WrongPKISub1Cert");
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_011() runs on SECC_Tester
                                                 system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_007("WrongPKICertChain");
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-----------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_012() runs on SECC_Tester
                                                 system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_008("DE1ABCD2EF357B");
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-----------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_013() runs on SECC_Tester
                                                 system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_008("Invalid_EMAID_");
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-----------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_014() runs on SECC_Tester
                                                 system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_009(cc_crt_contract_leaf_wrong_emaid);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_015() runs on SECC_Tester
                                                 system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_009(cc_crt_contract_leaf_invalid_emaid);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_016() runs on SECC_Tester
                                                 system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_010(cc_crt_contract_leaf_expired,
                                               fAILED_CertificateExpired);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateUpdate_017() runs on SECC_Tester
                                                 system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateUpdate_010(cc_crt_contract_leaf_invalid_content,
                                               m_CMN_failedResponseCodeList_001());
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```

D.2.2.10 SECC test cases for CertificateInstallation

```
module TestCases_SECC_CMN_PnC_CertificateInstallation {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_CertificateInstallation all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_Security all;

    testcase TC_SECC_CMN_VTB_CertificateInstallation_001() runs on SECC_Tester
                                                            system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------------
       if(preConVerdict==pass){
           f_SECC_CMN_TB_VTB_CertificateInstallation_001(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

   testcase TC_SECC_CMN_VTB_CertificateInstallation_002() runs on SECC_Tester
                                                              system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------------
      if(preConVerdict==pass){
          f_SECC_CMN_TB_VTB_CertificateInstallation_002();
      }else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

  testcase TC_SECC_CMN_VTB_CertificateInstallation_003() runs on SECC_Tester
                                                            system SystemSECC {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateInstallation_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_CMN_VTB_CertificateInstallation_004() runs on SECC_Tester
                                                        system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_SECC_CMN_TB_VTB_CertificateInstallation_004();
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_CMN_VTB_CertificateInstallation_005() runs on SECC_Tester
                                                        system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_CertificateInstallation_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

   testcase TC_SECC_CMN_VTB_CertificateInstallation_006() runs on SECC_Tester
                                                           system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_CertificateInstallation_006();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_SECC_CMN_VTB_CertificateInstallation_007() runs on SECC_Tester
                                                            system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_CertificateInstallation_007(cc_crt_oem_leaf_expired,
                                                          fAILED_CertificateExpired);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_SECC_CMN_VTB_CertificateInstallation_008() runs on SECC_Tester
                                                            system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_CertificateInstallation_007(cc_crt_oem_leaf_invalid_content,
                                                          m_CMN_failedResponseCodeList_001());
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```

D.2.2.11 SECC test cases for SessionStop

```
module TestCases_SECC_DC_CMN_SessionStop {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from TestBehavior_SECC_SessionStop all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
testcase TC_SECC_DC_VTB_SessionStop_001() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener, fail);
       }else {
           log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_SessionStop_002() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener, fail);
        }else {
            log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

testcase TC_SECC_DC_VTB_SessionStop_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_002();
       }else {
           log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionStop_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_SECC_CMN_TB_VTB_SessionStop_003(terminate);
       }else {
           log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_SessionStop_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_CMN_TB_VTB_SessionStop_004();
       }else {
           log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_SessionStop_006() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, false);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
          f_SECC_CMN_TB_VTB_SessionStop_005(terminate);
        }else {
            log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_DC_VTB_SessionStop_007() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_SECC_CMN_TB_VTB_SessionStop_002();
        }else {
            log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_DC_VTB_SessionStop_008() runs on SECC_Tester system SystemSECC {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_SessionStop_003(terminate);
           }else {
               log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_DC_VTB_SessionStop_009() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_SessionStop_004();
           }else {
               log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_SECC_AC_CMN_SessionStop {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from TestBehavior_SECC_SessionStop all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_SECC_AC_VTB_SessionStop_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener, fail);
           }else {
               log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

    testcase TC_SECC_AC_VTB_SessionStop_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_SessionStop_002();
           }else {
               log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_AC_VTB_SessionStop_003() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_CMN_TB_VTB_SessionStop_003(terminate);
           }else {
               log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_SECC_AC_VTB_SessionStop_004() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_SECC_CMN_TB_VTB_SessionStop_004();
            }else {
                log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```

D.2.2.12 SECC test cases for ChargingStatus

```
module TestCases_SECC_AC_PnC_ChargingStatus {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Pics_15118_2 all;
import from DataStructure_HAL_61851 all;
import from TestBehavior_SECC_ChargingStatus all;
import from Services_HAL_61851_1 all;
import from TestBehavior_SECC_CommonBehavior all;
import from PreConditions_SECC_15118_2 all;
import from PostConditions_SECC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;
import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;

testcase TC_SECC_AC_VTB_ChargingStatus_002()runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //------------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------

    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargingStatus_001(fail);
    }else {
           log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

  testcase TC_SECC_AC_VTB_ChargingStatus_007()runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      //------------------- Pre Conditions-------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
      preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

      //-------------- Test behavior-------------------------------------------------------------

      if(preConVerdict==pass){
          f_SECC_AC_TB_VTB_ChargingStatus_005();
      }else {
             log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions------------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
  }

  testcase TC_SECC_AC_VTB_ChargingStatus_009()runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      //------------------- Pre Conditions-------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
      preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

      //-------------- Test behavior-------------------------------------------------------------

      if(preConVerdict==pass){
          f_SECC_AC_TB_VTB_ChargingStatus_005();
      }else {
             log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions------------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

    testcase TC_SECC_AC_VTB_ChargingStatus_010()runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //------------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------

        if(preConVerdict==pass){
            f_SECC_AC_TB_VTB_ChargingStatus_001(fail);
        }else {
               log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

      testcase TC_SECC_AC_VTB_ChargingStatus_011()runs on SECC_Tester system SystemSECC {

          var HAL_61851_Listener v_HAL_61851_Listener;
          var verdicttype preConVerdict;

          //------------------- Pre Conditions-------------------------------------------------------
          f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
          preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

          //-------------- Test behavior-------------------------------------------------------------

          if(preConVerdict==pass){
              f_SECC_AC_TB_VTB_ChargingStatus_005();
          }else {
                 log("PreCondition was unsuccessful.");
          }

          //------------- Post Conditions------------------------------------------------------------
          f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
          f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
      }

      testcase TC_SECC_AC_VTB_ChargingStatus_012()runs on SECC_Tester system SystemSECC {

          var HAL_61851_Listener v_HAL_61851_Listener;
          var verdicttype preConVerdict;

          //------------------- Pre Conditions-------------------------------------------------------
          f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
          preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

          //-------------- Test behavior-------------------------------------------------------------

          if(preConVerdict==pass){
              f_SECC_AC_TB_VTB_ChargingStatus_005();
          }else {
                 log("PreCondition was unsuccessful.");
          }


          //------------- Post Conditions------------------------------------------------------------
          f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
          f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
      }
}



module TestCases_SECC_AC_CMN_ChargingStatus {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 import from Pics_15118_2 all;
 import from DataStructure_HAL_61851 all;
 import from TestBehavior_SECC_ChargingStatus all;
 import from Services_HAL_61851_1 all;
 import from TestBehavior_SECC_CommonBehavior all;
 import from PreConditions_SECC_15118_2 all;
 import from PostConditions_SECC_15118_2 all;
 import from ComponentsAndPorts all;
 import from Configurations_15118_2 all;
 import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;

testcase TC_SECC_AC_VTB_ChargingStatus_001()runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------

    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargingStatus_001(fail);
    }else {
           log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

 testcase TC_SECC_AC_VTB_ChargingStatus_003() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     //------------------- Pre Conditions-------------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

     //-------------- Test behavior-------------------------------------------------------------

     if(preConVerdict==pass){
         f_SECC_AC_TB_VTB_ChargingStatus_002();
     }else {
            log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
 }

testcase TC_SECC_AC_VTB_ChargingStatus_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //---- -------------- Pre Conditions-------------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------

    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_ChargingStatus_003();
    }else {
           log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

     testcase TC_SECC_AC_VTB_ChargingStatus_005() runs on SECC_Tester system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       //------------------- Pre Conditions-------------------------------------------------------
       f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
       preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

       //-------------- Test behavior-------------------------------------------------------------

           if(preConVerdict==pass){
           f_SECC_AC_TB_VTB_ChargingStatus_004();
       }else {
              log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions------------------------------------------------------------
       f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

     testcase TC_SECC_AC_VTB_ChargingStatus_006()runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
         preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

         //-------------- Test behavior-------------------------------------------------------------

         if(preConVerdict==pass){
             f_SECC_AC_TB_VTB_ChargingStatus_005();
         }else {
                log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions------------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
     }

     testcase TC_SECC_AC_VTB_ChargingStatus_008()runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
         preConVerdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

         //-------------- Test behavior-------------------------------------------------------------

         if(preConVerdict==pass){
             f_SECC_AC_TB_VTB_ChargingStatus_005();
         }else {
                log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions------------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
     }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

D.2.2.13 SECC test cases for MeteringReceipt

```
module TestCases_SECC_DC_PnC_MeteringReceipt {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_MeteringReceipt all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118_2 all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;

     testcase TC_SECC_DC_VTB_MeteringReceipt_001() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

          // ------------ Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

         // -------------- Test behavior-------------------------------------------------------------
         if ( preConVerdict == pass ) {
           f_SECC_DC_TB_VTB_MeteringReceipt_001(fail);
         }
         else {
           log ( "PreCondition was unsuccessful." );
         }

         // ------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
     }

        testcase TC_SECC_DC_VTB_MeteringReceipt_002() runs on SECC_Tester system SystemSECC {

           var HAL_61851_Listener v_HAL_61851_Listener;
           var verdicttype preConVerdict;

           // ------------ Pre Conditions-------------------------------------------------------
           f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
           preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

           //-------------- Test behavior-------------------------------------------------------------
           if ( preConVerdict == pass ) {
               f_SECC_CMN_TB_VTB_MeteringReceipt_001();
           }
           else {
               log ("PreCondition was unsuccessful.");
           }

           //------------- Post Conditions-------------------------------------------------------
           f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
           f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
        }

        testcase TC_SECC_DC_VTB_MeteringReceipt_003() runs on SECC_Tester system SystemSECC {

           var HAL_61851_Listener v_HAL_61851_Listener;
           var verdicttype preConVerdict;

           // ------------ Pre Conditions-------------------------------------------------------
           f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
           preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

           //-------------- Test behavior-------------------------------------------------------------
           if ( preConVerdict == pass ) {
               f_SECC_DC_TB_VTB_MeteringReceipt_002();
           }
           else {
               log ("PreCondition was unsuccessful.");
           }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //------------- Post Conditions-------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

testcase TC_SECC_DC_VTB_MeteringReceipt_004() runs on SECC_Tester system SystemSECC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // ------------ Pre Conditions-------------------------------------------------------
   f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
   preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

   //-------------- Test behavior-------------------------------------------------------------
   if ( preConVerdict == pass ) {
       f_SECC_DC_TB_VTB_MeteringReceipt_003();
   }
   else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------------
   f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

   testcase TC_SECC_DC_VTB_MeteringReceipt_005() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // ------------ Pre Conditions-------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
      preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

      //-------------- Test behavior-------------------------------------------------------------
      if ( preConVerdict == pass ) {
          f_SECC_CMN_TB_VTB_MeteringReceipt_002();
      }
      else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
   }

 testcase TC_SECC_DC_VTB_MeteringReceipt_006() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // ------------ Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
    preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------
    if ( preConVerdict == pass ) {
        f_SECC_DC_TB_VTB_MeteringReceipt_004();
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

 testcase TC_SECC_DC_VTB_MeteringReceipt_007() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        var verdicttype preConVerdict;

        // ------------ Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
        preConVerdict := f_SECC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if ( preConVerdict == pass ) {
            f_SECC_DC_TB_VTB_MeteringReceipt_005();
        }
        else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
     }
}



module TestCases_SECC_AC_PnC_MeteringReceipt {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_MeteringReceipt all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118_2 all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;

    testcase TC_SECC_AC_VTB_MeteringReceipt_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

         // ------------ Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        // -------------- Test behavior-------------------------------------------------------
        if ( preConVerdict == pass ) {
          f_SECC_AC_TB_VTB_MeteringReceipt_001(fail);
          }
          else {
          log ( "PreCondition was unsuccessful." );
          }

        // ------------- Post Conditions------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001           (v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

       testcase TC_SECC_AC_VTB_MeteringReceipt_002() runs on SECC_Tester system SystemSECC {

          var HAL_61851_Listener v_HAL_61851_Listener;
          var verdicttype preConVerdict;

          // ------------ Pre Conditions-------------------------------------------------------
          f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
          preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

          //-------------- Test behavior-------------------------------------------------------
          if ( preConVerdict == pass ) {
              f_SECC_CMN_TB_VTB_MeteringReceipt_001();
          }
          else {
              log ("PreCondition was unsuccessful.");
          }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //------------- Post Conditions-------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
     }

     testcase TC_SECC_AC_VTB_MeteringReceipt_003() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ------------ Pre Conditions-------------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        //-------------- Test behavior---------------------------------------------------------------
        if ( preConVerdict == pass ) {
            f_SECC_AC_TB_VTB_MeteringReceipt_002();
        }
        else {
            log ("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
     }

  testcase TC_SECC_AC_VTB_MeteringReceipt_004() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // ------------ Pre Conditions-------------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
     preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

     //-------------- Test behavior---------------------------------------------------------------
     if ( preConVerdict == pass ) {
         f_SECC_AC_TB_VTB_MeteringReceipt_003();
     }
     else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
  }

     testcase TC_SECC_AC_VTB_MeteringReceipt_005() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ------------ Pre Conditions-------------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
        preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        //-------------- Test behavior---------------------------------------------------------------
        if ( preConVerdict == pass ) {
            f_SECC_CMN_TB_VTB_MeteringReceipt_002();
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
     }

    testcase TC_SECC_AC_VTB_MeteringReceipt_006() runs on SECC_Tester system SystemSECC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        // ------------ Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
        preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if ( preConVerdict == pass ) {
            f_SECC_AC_TB_VTB_MeteringReceipt_004();
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
     }

     testcase TC_SECC_AC_VTB_MeteringReceipt_007() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // ------------ Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system );
        preConVerdict := f_SECC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if ( preConVerdict == pass ) {
            f_SECC_AC_TB_VTB_MeteringReceipt_005();
        }
        else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
     }
}
```

D.2.2.14 SECC test cases for CableCheck

```
module TestCases_SECC_DC_CMN_CableCheck {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_Authorization all;
    import from Services_HAL_61851_1 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_SECC_15118_2 all;
    import from TestBehavior_SECC_CableCheck all;
    import from PostConditions_SECC_15118_2 all;

    testcase TC_SECC_DC_VTB_CableCheck_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict == pass){
            f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, fail);
        }else {
          log("PreCondition was unsuccessful.");
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //------------- Post Conditions-------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_CableCheck_002() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------------
     if(preConVerdict == pass){
         f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, fail);
     }else {
       log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

testcase TC_SECC_DC_VTB_CableCheck_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------------
    if(preConVerdict == pass){
        f_SECC_DC_TB_VTB_CableCheck_002(v_HAL_61851_Listener);
    }else {
      log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_CableCheck_004() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------------
    if(preConVerdict == pass){
        f_SECC_DC_TB_VTB_CableCheck_003(v_HAL_61851_Listener);
    }else {
      log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_SECC_DC_VTB_CableCheck_005() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict == pass){
        f_SECC_DC_TB_VTB_CableCheck_004();
    }else {
      log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_CableCheck_006() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

     //-------------- Test behavior-------------------------------------------------------
     if(preConVerdict == pass){
         f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, fail);
     }else {
       log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

  testcase TC_SECC_DC_VTB_CableCheck_007() runs on SECC_Tester system SystemSECC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions------------------------------------------------------
      f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

      //-------------- Test behavior-------------------------------------------------------
      if(preConVerdict == pass){
          f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, fail);
      }else {
        log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions------------------------------------------------------
      f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_SECC_DC_VTB_CableCheck_008() runs on SECC_Tester system SystemSECC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions------------------------------------------------------
     f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

     //-------------- Test behavior-------------------------------------------------------
     if(preConVerdict == pass){
         f_SECC_DC_TB_VTB_CableCheck_005();
     }else {
       log("PreCondition was unsuccessful.");
     }


     //------------- Post Conditions------------------------------------------------------
     f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```

D.2.2.15 SECC test cases for PreCharge

```
module TestCases_SECC_DC_CMN_PreCharge {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_Authorization all;
    import from Configurations_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CableCheck all;
    import from PreConditions_SECC_15118_2  all;
    import from PostConditions_SECC_15118_2  all;
    import from TestBehavior_SECC_PreCharge all;
    import from ComponentsAndPorts all;

    testcase TC_SECC_DC_VTB_PreCharge_001() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if (preConVerdict == pass) {
             vc_EVTargetVoltage := par_EVBatteryVoltage;
             f_SECC_DC_TB_VTB_PreCharge_001(fail);
         } else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_DC_VTB_PreCharge_002() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if (preConVerdict == pass) {
             f_SECC_DC_TB_VTB_PreCharge_002();
         } else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_DC_VTB_PreCharge_003() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if (preConVerdict == pass) {
            f_SECC_DC_TB_VTB_PreCharge_003();
        } else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_DC_VTB_PreCharge_004() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
            f_SECC_DC_TB_VTB_PreCharge_004();
        } else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```

D.2.2.16 SECC test cases for CurrentDemand

```
module TestCases_SECC_DC_PnC_CurrentDemand {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_PowerDelivery all;
    import from TestBehavior_SECC_CurrentDemand all;
    import from TestBehavior_SECC_MeteringReceipt all;
    import from Configurations_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CableCheck all;
    import from PreConditions_SECC_15118_2  all;
    import from PostConditions_SECC_15118_2  all;
    import from ComponentsAndPorts all;

     testcase TC_SECC_DC_VTB_CurrentDemand_002() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         //------------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
         preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

         //-------------- Test behavior-------------------------------------------------------------
         if(preConVerdict==pass){
             f_SECC_DC_TB_VTB_CurrentDemand_001(fail);
         }else {
                log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
       }

     testcase TC_SECC_DC_VTB_CurrentDemand_007() runs on SECC_Tester system SystemSECC {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //------------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_CurrentDemand_005();
    }else {
           log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_CurrentDemand_009() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //------------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_CurrentDemand_005();
    }else {
           log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
}

testcase TC_SECC_DC_VTB_CurrentDemand_010() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //------------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_CurrentDemand_001(fail);
    }else {
           log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
 }

testcase TC_SECC_DC_VTB_CurrentDemand_011() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //------------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_CurrentDemand_005();
    }else {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

    testcase TC_SECC_DC_VTB_CurrentDemand_012() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //------------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //-------------- Test behavior---------------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_CurrentDemand_005();
        }else {
                log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }
}



module TestCases_SECC_DC_CMN_CurrentDemand {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_PowerDelivery all;
    import from TestBehavior_SECC_CurrentDemand all;
    import from TestBehavior_SECC_MeteringReceipt all;
    import from Configurations_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CableCheck all;
    import from PreConditions_SECC_15118_2  all;
    import from PostConditions_SECC_15118_2  all;
    import from ComponentsAndPorts all;

    testcase TC_SECC_DC_VTB_CurrentDemand_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //------------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //-------------- Test behavior---------------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_CurrentDemand_001(fail);
        }else {
                log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

      testcase TC_SECC_DC_VTB_CurrentDemand_003() runs on SECC_Tester system SystemSECC {

          var HAL_61851_Listener v_HAL_61851_Listener;
          var verdicttype preConVerdict;

          //------------------- Pre Conditions-------------------------------------------------------
          f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_CurrentDemand_002();
        }else {
               log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

    testcase TC_SECC_DC_VTB_CurrentDemand_004() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //------------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_CurrentDemand_003();
        }else {
               log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

    testcase TC_SECC_DC_VTB_CurrentDemand_005() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //------------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_CurrentDemand_004();
        }else {
               log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
    }

    testcase TC_SECC_DC_VTB_CurrentDemand_006() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //------------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_CurrentDemand_005();
        }else {
               log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     }

     testcase TC_SECC_DC_VTB_CurrentDemand_008() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         //------------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
         preConVerdict := f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

         //-------------- Test behavior---------------------------------------------------------------
         if(preConVerdict==pass){
             f_SECC_DC_TB_VTB_CurrentDemand_005();
         }else {
                log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-------------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system)
     }
}
```

D.2.2.17 SECC test cases for WeldingDetection

```
module TestCases_SECC_DC_CMN_WeldingDetection {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from TestBehavior_SECC_WeldingDetection all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_SECC_DC_VTB_WeldingDetection_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

        //-------------- Test behavior---------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_WeldingDetection_001(fail);
           }else {
               log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_DC_VTB_WeldingDetection_002() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

        //-------------- Test behavior---------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_WeldingDetection_002();
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }else {
                log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_DC_VTB_WeldingDetection_003() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_WeldingDetection_003();
            }else {
                log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_SECC_DC_VTB_WeldingDetection_004() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_WeldingDetection_004();
            }else {
                log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_SECC_DC_VTB_WeldingDetection_005() runs on SECC_Tester system SystemSECC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, false);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_SECC_DC_TB_VTB_WeldingDetection_005();
             }else {
                 log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-------------------------------------------------------
         f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

D.3 EVCC test cases

D.3.1 V2G protocol handshake

D.3.1.1 EVCC test cases for SupportedAppProtocol

```
module TestCases_EVCC_CMN_EIM_SupportedAppProtocol {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from Timer_15118_2 all;
    import from TestBehavior_EVCC_SupportedAppProtocol all;

     testcase TC_EVCC_CMN_VTB_SupportedAppProtocol_004() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_CMN_PR_SDP_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_PR_InitConfiguration_002(system, fail);
             f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001(fail);
          }
          else {
             log("PreCondition was unsuccessful.");
          }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_EVCC_CMN_VTB_SupportedAppProtocol_005() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_CMN_PR_SDP_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_PR_InitConfiguration_002(system, fail);
             f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001(fail);
          }
          else {
             log("PreCondition was unsuccessful.");
          }


         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
module TestCases_EVCC_CMN_CMN_SupportedAppProtocol {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from Timer_15118_2 all;
    import from TestBehavior_EVCC_SupportedAppProtocol all;

    testcase TC_EVCC_CMN_VTB_SupportedAppProtocol_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_SDP_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_PR_InitConfiguration_002(system, fail);
            f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001(fail);
         }
         else {
            log("PreCondition was unsuccessful.");
         }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_SupportedAppProtocol_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SupportedAppProtocol_002();
         }
         else {
            log("PreCondition was unsuccessful.");
         }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_SupportedAppProtocol_003() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SupportedAppProtocol_003();
         }
         else {
            log("PreCondition was unsuccessful.");
         }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```

D.3.2 V2G messages

D.3.2.1 EVCC test cases for SessionSetup

```
module TestCases_EVCC_DC_CMN_SessionSetup {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_SessionSetup all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851_1 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_EVCC_DC_VTB_SessionSetup_001() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(tsInitiated);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                             v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_EVCC_DC_VTB_SessionSetup_002() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(sutInitiated);
         if(preConVerdict == pass) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                             v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
        }

        if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_SessionSetup_003() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
        if(preConVerdict == pass) {
            preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                             v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            var float v_dataLinkprocessing;
            if(tc_CMN_ProcessingTime_Timer.running) {
                v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
            } else {
                v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
            }
            preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                             v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_SessionSetup_004() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                            v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                            v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
           preConVerdict :=
           f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}



module TestCases_EVCC_CMN_CMN_SessionSetup {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_SessionSetup all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851_1 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;

    testcase TC_EVCC_CMN_VTB_SessionSetup_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_SessionSetup_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SessionSetup_002() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_SessionSetup_002(fAILED);
   }else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);

  }

  testcase TC_EVCC_CMN_VTB_SessionSetup_003() runs on EVCC_Tester system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
          f_EVCC_CMN_TB_VTB_SessionSetup_002(fAILED_SequenceError);
      }else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SessionSetup_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_SessionSetup_005() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SessionSetup_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_005(e_OpenCircuit);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SessionSetup_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_005(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_SessionSetup_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }


    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     }
}



module TestCases_EVCC_AC_CMN_SessionSetup {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_SessionSetup all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851_1 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_EVCC_AC_VTB_SessionSetup_001() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(tsInitiated);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_EVCC_AC_VTB_SessionSetup_002() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(sutInitiated);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
         }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
       preConVerdict :=
       f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions----------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionSetup_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-----------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
           v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
       } else {
           v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
       }
       preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                        v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
       preConVerdict :=
       f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions----------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionSetup_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-----------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if(tc_CMN_ProcessingTime_Timer.running) {
                v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
            } else {
                v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
            }
            preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                             v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior-------------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_SessionSetup_006(fail);
         }else {
             log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions------------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}
```

D.3.2.2 EVCC test cases for ServiceDiscovery

```
module TestCases_EVCC_CMN_CMN_ServiceDiscovery {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from Configurations_15118_2 all;

   testcase TC_EVCC_CMN_VTB_ServiceDiscovery_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDiscovery_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

   testcase TC_EVCC_CMN_VTB_ServiceDiscovery_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDiscovery_002(fAILED);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_ServiceDiscovery_003() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDiscovery_002(fAILED_SequenceError);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_ServiceDiscovery_004() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDiscovery_002(fAILED_UnknownSession);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_ServiceDiscovery_005() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDiscovery_003();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     testcase TC_EVCC_CMN_VTB_ServiceDiscovery_006() runs on EVCC_Tester system SystemEVCC {

          var HAL_61851_Listener v_HAL_61851_Listener;
          var verdicttype preConVerdict;

          // -------------- Pre Conditions-------------------------------------------------------
          f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
          preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

          //-------------- Test behavior--------------------------------------------------------
          if(preConVerdict==pass){
              f_EVCC_CMN_TB_VTB_ServiceDiscovery_004(e_OpenCircuit);
          }else {
              log("PreCondition was unsuccessful.");
          }

          //------------- Post Conditions-----------------------------------------------------
          f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
          f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

     testcase TC_EVCC_CMN_VTB_ServiceDiscovery_007() runs on EVCC_Tester system SystemEVCC {

          var HAL_61851_Listener v_HAL_61851_Listener;
          var verdicttype preConVerdict;

          // -------------- Pre Conditions-------------------------------------------------------
          f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
          preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

          //-------------- Test behavior--------------------------------------------------------
          if(preConVerdict==pass){
              f_EVCC_CMN_TB_VTB_ServiceDiscovery_004(e_NegVolt12);
          }else {
              log("PreCondition was unsuccessful.");
          }

          //------------- Post Conditions-----------------------------------------------------
          f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
          f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

     testcase TC_EVCC_CMN_VTB_ServiceDiscovery_008() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDiscovery_005();
         }
         else {
            log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-----------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```

D.3.2.3 EVCC test cases for ServiceDetail

```
module TestCases_EVCC_CMN_PnC_ServiceDetail {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;
import from PreConditions_EVCC_15118_2 all;
import from PostConditions_EVCC_15118_2 all;
import from TestBehavior_EVCC_ServiceDetail all;

 testcase TC_EVCC_CMN_VTB_ServiceDetail_001() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_ServiceDetail_001(fail);
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_CMN_VTB_ServiceDetail_004() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED);
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_CMN_VTB_ServiceDetail_006() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_SequenceError);
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_CMN_VTB_ServiceDetail_008() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_ServiceIDInvalid);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_ServiceDetail_016() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetail_006(e_OpenCircuit);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_ServiceDetail_018() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetail_006(e_NegVolt12);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_EVCC_CMN_EIM_ServiceDetail {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_ServiceDetail all;

    testcase TC_EVCC_CMN_VTB_ServiceDetail_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetail_002(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_003(fAILED_ServiceIDInvalid);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetail_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetail_006(e_OpenCircuit);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    testcase TC_EVCC_CMN_VTB_ServiceDetail_017() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetail_006(e_NegVolt12);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-----------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```

D.3.2.4 EVCC test cases for ServiceDetailAndPaymentSelection

```
module TestCases_EVCC_DC_PnC_ServiceDetailAndPaymentSelection {

   import from ComponentsAndPorts all;
   import from Configurations_15118_2 all;
   import from PreConditions_EVCC_15118_2 all;
   import from PostConditions_EVCC_15118_2 all;
   import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
   import from Timer_15118 all;
   import from TestBehavior_EVCC_CommonBehavior all;
   import from Services_HAL_61851_1 all;
   import from Pics_15118 all;
   import from Pixit_15118 all;
   import from Timer_15118_2 all;
   import from Timer_15118_3 all;
   import from LibFunctions_15118_2 all;
   import from DataStructure_PICS_15118 all;

     testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001() runs on EVCC_Tester
                                                                    system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(tsInitiated);
        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                            v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
           preConVerdict :=
           f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003() runs on EVCC_Tester
                                                               system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005() runs on EVCC_Tester
                                                               system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
         }else {
            log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

      testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007() runs on EVCC_Tester
                                                                     system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                             v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            var float v_dataLinkprocessing;
            if(tc_CMN_ProcessingTime_Timer.running) {
                v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
            } else {
                v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
            }
            preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                               v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
         }else {
            log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}



module TestCases_EVCC_DC_EIM_ServiceDetailAndPaymentSelection {

   import from ComponentsAndPorts all;
   import from Configurations_15118_2 all;
   import from PreConditions_EVCC_15118_2 all;
   import from PostConditions_EVCC_15118_2 all;
   import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
   import from Timer_15118 all;
   import from TestBehavior_EVCC_CommonBehavior all;
   import from Services_HAL_61851_1 all;
   import from Pics_15118 all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Pixit_15118 all;
import from Timer_15118_2 all;
import from Timer_15118_3 all;
import from LibFunctions_15118_2 all;
import from DataStructure_PICS_15118 all;

  testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002() runs on EVCC_Tester
                                                                 system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_checkWakeUpConfig(tsInitiated);
     if(preConVerdict == pass) {
        preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                         v_HAL_61851_Listener);
     }

     if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
     }

     if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
     }

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
     }else {
        log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

  testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004() runs on EVCC_Tester
                                                                 system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_checkWakeUpConfig(sutInitiated);
     if(preConVerdict == pass) {
        preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                         v_HAL_61851_Listener);
     }

     if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
     }

     if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
     }

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
     }else {
        log("PreCondition was unsuccessful.");
     }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   //------------- Post Conditions-------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006() runs on EVCC_Tester
                                                               system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008() runs on EVCC_Tester
                                                               system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
           preConVerdict :=
           f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
        }else {
           log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-----------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}



module TestCases_EVCC_CMN_PnC_ServiceDetailAndPaymentSelection {

   import from ComponentsAndPorts all;
   import from Configurations_15118_2 all;
   import from PreConditions_EVCC_15118_2 all;
   import from PostConditions_EVCC_15118_2 all;
   import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
   import from Timer_15118 all;
   import from TestBehavior_EVCC_CommonBehavior all;
   import from Services_HAL_61851_1 all;
   import from Pics_15118 all;
   import from Pixit_15118 all;
   import from Timer_15118_2 all;

   testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_001() runs on EVCC_Tester
                                                                      system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-----------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

       //-------------- Test behavior-------------------------------------------------------------
       if(preConVerdict==pass){
           f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_001(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }


       //------------- Post Conditions-----------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }
}



module TestCases_EVCC_CMN_EIM_ServiceDetailAndPaymentSelection {

   import from ComponentsAndPorts all;
   import from Configurations_15118_2 all;
   import from PreConditions_EVCC_15118_2 all;
   import from PostConditions_EVCC_15118_2 all;
   import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
   import from Timer_15118 all;
   import from TestBehavior_EVCC_CommonBehavior all;
   import from Services_HAL_61851_1 all;
   import from Pics_15118 all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   import from Pixit_15118 all;
   import from Timer_15118_2 all;

   testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_002() runs on EVCC_Tester
                                                                     system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict==pass){
           f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_002(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }


       //------------- Post Conditions-------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }
}



module TestCases_EVCC_CMN_CMN_ServiceDetailAndPaymentSelection {

   import from ComponentsAndPorts all;
   import from Configurations_15118_2 all;
   import from PreConditions_EVCC_15118_2 all;
   import from PostConditions_EVCC_15118_2 all;
   import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
   import from Timer_15118 all;
   import from TestBehavior_EVCC_CommonBehavior all;
   import from Services_HAL_61851_1 all;
   import from Pics_15118 all;
   import from Pixit_15118 all;
   import from Timer_15118_2 all;

    testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003() runs on EVCC_Tester
                                                                     system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(fAILED);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004() runs on EVCC_Tester
                                                                     system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005() runs on EVCC_Tester
                                                                system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006() runs on EVCC_Tester
                                                                system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(fAILED_PaymentSelectionInvalid);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007() runs on EVCC_Tester
                                                                system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(fAILED_ServiceSelectionInvalid);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008() runs on EVCC_Tester
                                                                system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(fAILED_NoChargeServiceSelected);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009() runs on EVCC_Tester
                                                                  system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict :=
     f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_006();
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010() runs on EVCC_Tester
                                                                 system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict :=
     f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_007();
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011() runs on EVCC_Tester
                                                                    system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_008(e_OpenCircuit);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-----------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012() runs on EVCC_Tester
                                                                    system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_008(e_NegVolt12);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-----------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_EVCC_AC_PnC_ServiceDetailAndPaymentSelection {

   import from ComponentsAndPorts all;
   import from Configurations_15118_2 all;
   import from PreConditions_EVCC_15118_2 all;
   import from PostConditions_EVCC_15118_2 all;
   import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
   import from Timer_15118 all;
   import from TestBehavior_EVCC_CommonBehavior all;
   import from Services_HAL_61851_1 all;
   import from Pics_15118 all;
   import from Pixit_15118 all;
   import from Timer_15118_2 all;
   import from Timer_15118_3 all;
   import from LibFunctions_15118_2 all;
   import from DataStructure_PICS_15118 all;

     testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001() runs on EVCC_Tester
                                                                    system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003() runs on EVCC_Tester
                                                               system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005() runs on EVCC_Tester
                                                               system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007() runs on EVCC_Tester
                                                               system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(fail);
   }else {
      log("PreCondition was unsuccessful.");
   }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}



module TestCases_EVCC_AC_EIM_ServiceDetailAndPaymentSelection {

   import from ComponentsAndPorts all;
   import from Configurations_15118_2 all;
   import from PreConditions_EVCC_15118_2 all;
   import from PostConditions_EVCC_15118_2 all;
   import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
   import from Timer_15118 all;
   import from TestBehavior_EVCC_CommonBehavior all;
   import from Services_HAL_61851_1 all;
   import from Pics_15118 all;
   import from Pixit_15118 all;
   import from Timer_15118_2 all;
   import from Timer_15118_3 all;
   import from LibFunctions_15118_2 all;
   import from DataStructure_PICS_15118 all;

      testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002() runs on EVCC_Tester
                                                                     system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(tsInitiated);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
         }else {
            log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

      testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004() runs on EVCC_Tester
                                                                     system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(sutInitiated);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
         }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006() runs on EVCC_Tester
                                                               system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008() runs on EVCC_Tester
                                                               system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           var float v_dataLinkprocessing;
           if(tc_CMN_ProcessingTime_Timer.running) {
               v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
           } else {
               v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
           }
           preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                              v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
           preConVerdict :=
           f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(fail);
        }else {
           log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-----------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```

D.3.2.5 EVCC test cases for PaymentServiceSelection

```
module TestCases_EVCC_DC_PnC_PaymentServiceSelection {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_PaymentServiceSelection all;
    import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
    import from Timer_15118 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_EVCC_DC_VTB_PaymentServiceSelection_001() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(tsInitiated);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                             v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
     }

     if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
     }

     //-------------- Test behavior-------------------------------------------------------
     if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
     }else {
        log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_003() runs on EVCC_Tester
                                                        system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_005() runs on EVCC_Tester
                                                       system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                        v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
       preConVerdict :=
       f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
    }else {
       log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_007() runs on EVCC_Tester
                                                      system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_009() runs on EVCC_Tester
                                                      system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                        v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
        } else {
            v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
        }
        preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                         v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
     }

     if(preConVerdict == pass) {
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
     }

     //-------------- Test behavior-------------------------------------------------------------
     if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
     }else {
        log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions------------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_011() runs on EVCC_Tester
                                                      system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                         v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_013() runs on EVCC_Tester
                                                      system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                        v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
           v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
       } else {
           v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
       }
       preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                        v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
       preConVerdict :=
       f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
    }else {
       log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_015() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
   }else {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}



module TestCases_EVCC_DC_EIM_PaymentServiceSelection {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_PaymentServiceSelection all;
    import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
    import from Timer_15118 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_EVCC_DC_VTB_PaymentServiceSelection_002() runs on EVCC_Tester
                                                           system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(tsInitiated);
        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                            v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
           preConVerdict :=
           f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
           f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
        }else {
           log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_EVCC_DC_VTB_PaymentServiceSelection_004() runs on EVCC_Tester
                                                            system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions----------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_006() runs on EVCC_Tester
                                                      system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-----------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
      f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions----------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_008() runs on EVCC_Tester
                                                      system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-----------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_010() runs on EVCC_Tester
                                                      system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
      f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_012() runs on EVCC_Tester
                                                      system SystemEVCC {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PaymentServiceSelection_014() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
         if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
         }else {
            log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

      testcase TC_EVCC_DC_VTB_PaymentServiceSelection_016() runs on EVCC_Tester
                                                            system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                             v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            var float v_dataLinkprocessing;
            if(tc_CMN_ProcessingTime_Timer.running) {
                v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
            } else {
                v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
            }
            preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                               v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
         }else {
            log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
   }



   module TestCases_EVCC_CMN_PnC_PaymentServiceSelection {

       import from ComponentsAndPorts all;
       import from Configurations_15118_2 all;
       import from PreConditions_EVCC_15118_2 all;
       import from PostConditions_EVCC_15118_2 all;
       import from TestBehavior_EVCC_PaymentServiceSelection all;
       import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
       import from Timer_15118 all;
       import from TestBehavior_EVCC_CommonBehavior all;
       import from Services_HAL_61851_1 all;
       import from Pics_15118 all;
       import from Pixit_15118 all;
       import from Timer_15118_2 all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    testcase TC_EVCC_CMN_VTB_PaymentServiceSelection_001() runs on EVCC_Tester
                                                            system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_003();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_CMN_VTB_PaymentServiceSelection_003() runs on EVCC_Tester
                                                             system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_PaymentServiceSelection_001();
         }else {
             log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}



module TestCases_EVCC_CMN_EIM_PaymentServiceSelection {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_PaymentServiceSelection all;
    import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
    import from Timer_15118 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;

     testcase TC_EVCC_CMN_VTB_PaymentServiceSelection_002() runs on EVCC_Tester
                                                             system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_004();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_CMN_VTB_PaymentServiceSelection_004() runs on EVCC_Tester
                                                              system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_PaymentServiceSelection_002();
         }else {
             log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}



module TestCases_EVCC_AC_PnC_PaymentServiceSelection {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_PaymentServiceSelection all;
    import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
    import from Timer_15118 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_EVCC_AC_VTB_PaymentServiceSelection_002() runs on EVCC_Tester
                                                              system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(tsInitiated);
        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
        }

        if(preConVerdict == pass) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        preConVerdict :=
        f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
     }

     //-------------- Test behavior-------------------------------------------------------------
     if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
     }else {
        log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions------------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_004() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_006() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiated);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
       preConVerdict :=
       f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
    }else {
       log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_008() runs on EVCC_Tester
                                                      system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-----------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_010() runs on EVCC_Tester
                                                       system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-----------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
           v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
       } else {
           v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
       }
       preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                        v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
       preConVerdict :=
       f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
    }else {
       log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_012() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_014() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            var float v_dataLinkprocessing;
            if(tc_CMN_ProcessingTime_Timer.running) {
                v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
            } else {
                v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
            }
            preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                             v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009();
         }else {
            log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

      testcase TC_EVCC_AC_VTB_PaymentServiceSelection_016() runs on EVCC_Tester
                                                            system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            var float v_dataLinkprocessing;
            if(tc_CMN_ProcessingTime_Timer.running) {
                v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
            } else {
                v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
            }
            preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                             v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003();
         }else {
            log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
module TestCases_EVCC_AC_EIM_PaymentServiceSelection {

    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_PaymentServiceSelection all;
    import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
    import from Timer_15118 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

     testcase TC_EVCC_AC_VTB_PaymentServiceSelection_001() runs on EVCC_Tester
                                                           system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_AC_TB_VTB_PaymentServiceSelection_001(cc_dutyCycle_32A);
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_EVCC_AC_VTB_PaymentServiceSelection_003() runs on EVCC_Tester
                                                           system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_checkWakeUpConfig(tsInitiated);
        if(preConVerdict == pass) {
           preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
        }

        if(preConVerdict == pass) {
           map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
           preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
        }

        if(preConVerdict == pass) {
           unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
           preConVerdict :=
           f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
        }

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
           f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
        }else {
           log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_005() runs on EVCC_Tester
                                                      system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_007() runs on EVCC_Tester
                                                      system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
      f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
testcase TC_EVCC_AC_VTB_PaymentServiceSelection_009() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_011() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------------
   if(preConVerdict==pass){
      f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
   }else {
      log("PreCondition was unsuccessful.");
   }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_013() runs on EVCC_Tester
                                                      system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions--------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior---------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_PaymentServiceSelection_015() runs on EVCC_Tester
                                                      system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions--------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010();
         }else {
            log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions----------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

      testcase TC_EVCC_AC_VTB_PaymentServiceSelection_017() runs on EVCC_Tester
                                                            system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-----------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
         if(preConVerdict == pass) {
            preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
         }

         if(preConVerdict == pass) {
            map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
            var float v_dataLinkprocessing;
            if(tc_CMN_ProcessingTime_Timer.running) {
                v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
            } else {
                v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
            }
            preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                                v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004();
         }else {
            log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions----------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}
```

D.3.2.6 EVCC test cases for CertificateInstallation

```
module TestCases_EVCC_CMN_PnC_CertificateInstallation {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_CertificateInstallation all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 import from ComponentsAndPorts all;
 import from Configurations_15118_2 all;

testcase TC_EVCC_CMN_VTB_CertificateInstallation_001() runs on EVCC_Tester
                                                          system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateInstallation_001(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateInstallation_002() runs on EVCC_Tester
                                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateInstallation_003() runs on EVCC_Tester
                                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateInstallation_004() runs on EVCC_Tester
                                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateInstallation_005() runs on EVCC_Tester
                                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED_SignatureError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateInstallation_006() runs on EVCC_Tester
                                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED_CertificateExpired);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateInstallation_007() runs on EVCC_Tester
                                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED_CertificateRevoked);
    }else {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateInstallation_008() runs on EVCC_Tester
                                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateInstallation_002(fAILED_NoCertificateAvailable);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateInstallation_009() runs on EVCC_Tester
                                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateInstallation_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateInstallation_010() runs on EVCC_Tester
                                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateInstallation_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateInstallation_011() runs on EVCC_Tester
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                        system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_CertificateInstallation_005(e_OpenCircuit);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_CMN_VTB_CertificateInstallation_012() runs on EVCC_Tester
                                                        system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_CertificateInstallation_005(e_NegVolt12);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```

D.3.2.7 EVCC test cases for CertificateUpdate

```
module TestCases_EVCC_CMN_PnC_CertificateUpdate {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_CertificateUpdate all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

   testcase TC_EVCC_CMN_VTB_CertificateUpdate_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_CertificateUpdate_001(fail);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_CertificateUpdate_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_CertificateUpdate_003() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_SequenceError);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_CertificateUpdate_004() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_UnknownSession);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_CertificateUpdate_005() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_SignatureError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateUpdate_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_CertificateExpired);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateUpdate_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_CertificateRevoked);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateUpdate_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_NoCertificateAvailable);
    }else {
        log("PreCondition was unsuccessful.");
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateUpdate_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_ContractCanceled);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateUpdate_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateUpdate_002(fAILED_ChallengeInvalid);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateUpdate_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
     f_EVCC_CMN_TB_VTB_CertificateUpdate_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateUpdate_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_CertificateUpdate_004();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_CMN_VTB_CertificateUpdate_013() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
          f_EVCC_CMN_TB_VTB_CertificateUpdate_005(e_OpenCircuit);
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_CMN_VTB_CertificateUpdate_014() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
          f_EVCC_CMN_TB_VTB_CertificateUpdate_005(e_NegVolt12);
         }else {
             log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```

D.3.2.8 EVCC test cases for CertificateHandlingAndPaymentDetails

```
module TestCases_EVCC_CMN_PnC_CertificateHandlingAndPaymentDetails {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_CertificateHandlingAndPaymentDetails all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_001() runs on EVCC_Tester
                                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict :=
     f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_001(fail);
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002() runs on EVCC_Tester
                                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002(fAILED);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003() runs on EVCC_Tester
                                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004() runs on EVCC_Tester
                                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005() runs on EVCC_Tester
                                                                      system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_003();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006() runs on EVCC_Tester
                                                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007() runs on EVCC_Tester
                                                                     system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_005(e_OpenCircuit);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008() runs on EVCC_Tester
                                                                          system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_005(e_NegVolt12);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```

D.3.2.9 EVCC test cases for PaymentDetails

```
module TestCases_EVCC_CMN_PnC_PaymentDetails {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_PaymentDetails all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

   testcase TC_EVCC_CMN_VTB_PaymentDetails_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_PaymentDetails_001();
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   testcase TC_EVCC_CMN_VTB_PaymentDetails_002() runs on EVCC_Tester system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_EVCC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
          f_EVCC_CMN_TB_VTB_PaymentDetails_002();
      }else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

   testcase TC_EVCC_CMN_VTB_PaymentDetails_003() runs on EVCC_Tester system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict :=
      f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
          f_EVCC_CMN_TB_VTB_PaymentDetails_003();
      }else {
          log("PreCondition was unsuccessful.");
      }


      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }
}
```

D.3.2.10 EVCC test cases for Authorization

```
module TestCases_EVCC_CMN_PnC_Authorization {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_Authorization all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

   testcase TC_EVCC_CMN_VTB_Authorization_001() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict :=
       f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict==pass){
           f_EVCC_CMN_TB_VTB_Authorization_001(fail);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_CMN_VTB_Authorization_007() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_Authorization_003(fAILED_SignatureError);
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

    testcase TC_EVCC_CMN_VTB_Authorization_010() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_Authorization_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_CMN_VTB_Authorization_011() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict :=
        f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_Authorization_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
module TestCases_EVCC_CMN_EIM_Authorization {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_Authorization all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

   testcase TC_EVCC_CMN_VTB_Authorization_002() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict :=
       f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict==pass){
           f_EVCC_CMN_TB_VTB_Authorization_001(fail);
       }else {
           log("PreCondition was unsuccessful.");
       }


       //------------- Post Conditions-------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }
}



module TestCases_EVCC_CMN_CMN_Authorization {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_Authorization all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

   testcase TC_EVCC_CMN_VTB_Authorization_003() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict==pass){
           f_EVCC_CMN_TB_VTB_Authorization_003(fAILED);
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

   testcase TC_EVCC_CMN_VTB_Authorization_004() runs on EVCC_Tester system SystemEVCC {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_003(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_003(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_003(fAILED_ChallengeInvalid);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_004();
    }else {
        log("PreCondition was unsuccessful.");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_CMN_VTB_Authorization_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_Authorization_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_CMN_VTB_Authorization_012() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_CMN_TB_VTB_Authorization_006();
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

  testcase TC_EVCC_CMN_VTB_Authorization_013() runs on EVCC_Tester system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_EVCC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
          f_EVCC_CMN_TB_VTB_Authorization_007(e_OpenCircuit);
      }else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
  }

  testcase TC_EVCC_CMN_VTB_Authorization_014() runs on EVCC_Tester system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_Authorization_002(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_Authorization_007(e_NegVolt12);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```

D.3.2.11 EVCC test cases for ChargeParameterDiscovery

```
module TestCases_EVCC_DC_CMN_ChargeParameterDiscovery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargeParameterDiscovery all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

    testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_001() runs on EVCC_Tester
                                                            system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_002() runs on EVCC_Tester
                                                            system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  //-------------- Test behavior-------------------------------------------------------------
  if(preConVerdict==pass){
      f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001(fAILED);
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_003() runs on EVCC_Tester
                                                         system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions------------------------------------------------------------
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);


  //-------------- Test behavior-------------------------------------------------------------
  if(preConVerdict==pass){
      f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001(fAILED_SequenceError);
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_004() runs on EVCC_Tester
                                                         system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions------------------------------------------------------------
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);


  //-------------- Test behavior-------------------------------------------------------------
  if(preConVerdict==pass){
      f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001(fAILED_UnknownSession);
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions------------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_005() runs on EVCC_Tester
                                                         system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions------------------------------------------------------------
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);


  //-------------- Test behavior-------------------------------------------------------------
  if(preConVerdict==pass){
      f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001(fAILED_WrongEnergyTransferMode);
  }else {
      log("PreCondition was unsuccessful.");
  }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  //------------- Post Conditions-------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_006() runs on EVCC_Tester
                                                        system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions--------------------------------------------------------
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);


  //-------------- Test behavior---------------------------------------------------------
  if(preConVerdict==pass){
      f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001(fAILED_WrongChargeParameter);
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_007() runs on EVCC_Tester
                                                        system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions--------------------------------------------------------
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

  //-------------- Test behavior---------------------------------------------------------
  if(preConVerdict==pass){
      f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_002();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_008() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_003(v_HAL_61851_Listener, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_009() runs on EVCC_Tester
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_004(e_OpenCircuit);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_010() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_004(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_011() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_003(v_HAL_61851_Listener, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_012() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_013() runs on EVCC_Tester
                                                       system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_006(v_HAL_61851_Listener);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_014() runs on EVCC_Tester
                                                       system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_006(v_HAL_61851_Listener);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_015() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      if(preConVerdict == pass) {
         map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
         preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
      }

      if(preConVerdict == pass) {
         unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
         preConVerdict :=
         f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
      }

      //-------------- Test behavior-------------------------------------------------------
      if(preConVerdict==pass){
          f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
      }else {
         log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions----------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

   testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_016() runs on EVCC_Tester
                                                          system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-----------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_checkWakeUpConfig(tsInitiated);
      if(preConVerdict == pass) {
         preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                          v_HAL_61851_Listener);
      }

      if(preConVerdict == pass) {
         map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
         preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
      }

      if(preConVerdict == pass) {
         unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
         preConVerdict :=
         f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
      }

      //-------------- Test behavior-------------------------------------------------------
      if(preConVerdict==pass){
          f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
      }else {
         log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions----------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

   testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_017() runs on EVCC_Tester
                                                          system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-----------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_checkWakeUpConfig(tsInitiated);
      if(preConVerdict == pass) {
         preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                          v_HAL_61851_Listener);
      }

      if(preConVerdict == pass) {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_018() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_019() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions----------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_020() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions----------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_021() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_022() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_023() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_024() runs on EVCC_Tester
                                                        system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_025() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions--------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior---------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_ChargeParameterDiscovery_026() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions--------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(
                       v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                                v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
         }

         if(preConVerdict == pass) {
            unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
            preConVerdict :=
            f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
         }

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
         }else {
            log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions----------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}



module TestCases_EVCC_CMN_CMN_ChargeParameterDiscovery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargeParameterDiscovery all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;

     testcase TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001() runs on EVCC_Tester
                                                              system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-----------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

       //-------------- Test behavior-------------------------------------------------------
       if(preConVerdict==pass){
           f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_002();
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions----------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002() runs on EVCC_Tester
                                                             system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-----------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
           f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_003();
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }
}



module TestCases_EVCC_AC_CMN_ChargeParameterDiscovery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargeParameterDiscovery all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_PICS_15118 all;

   testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_001() runs on EVCC_Tester
                                                            system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_001(fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
      }

     testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_002() runs on EVCC_Tester
                                                            system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);


        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001(fAILED);
        }else {
            log("PreCondition was unsuccessful.");
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
  //------------- Post Conditions-------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_003() runs on EVCC_Tester
                                                         system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions--------------------------------------------------------
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);


  //-------------- Test behavior---------------------------------------------------------
  if(preConVerdict==pass){
      f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001(fAILED_SequenceError);
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_004() runs on EVCC_Tester
                                                         system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions--------------------------------------------------------
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);


  //-------------- Test behavior---------------------------------------------------------
  if(preConVerdict==pass){
      f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001(fAILED_UnknownSession);
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_005() runs on EVCC_Tester
                                                         system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions--------------------------------------------------------
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);


  //-------------- Test behavior---------------------------------------------------------
  if(preConVerdict==pass){
      f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001(fAILED_WrongEnergyTransferMode);
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_006() runs on EVCC_Tester
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                        system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);


  //-------------- Test behavior--------------------------------------------------------
  if(preConVerdict==pass){
      f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001(fAILED_WrongChargeParameter);
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_007() runs on EVCC_Tester
                                                        system SystemEVCC {

  var HAL_61851_Listener v_HAL_61851_Listener;
  var verdicttype preConVerdict;

  // -------------- Pre Conditions-------------------------------------------------------
  f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
  preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

  //-------------- Test behavior--------------------------------------------------------
  if(preConVerdict==pass){
      f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_002();
  }else {
      log("PreCondition was unsuccessful.");
  }

  //------------- Post Conditions-------------------------------------------------------
  f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
  f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_008() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_003(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_009() runs on EVCC_Tester
                                                       system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    preConVerdict :=
    f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_004(e_OpenCircuit);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_010() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_004(e_NegVolt12);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_011() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_003(fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_012() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_005();
    }else {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_013() runs on EVCC_Tester
                                                       system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_006();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_014() runs on EVCC_Tester
                                                       system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_006();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_015() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_016() runs on EVCC_Tester
                                                        system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_017() runs on EVCC_Tester
                                                        system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(tsInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PICS_CMN_CMN_WakeUp);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_018() runs on EVCC_Tester
                                                        system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_019() runs on EVCC_Tester
                                                        system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_020() runs on EVCC_Tester
                                                        system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiated);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      preConVerdict := f_EVCC_startSleepingPhase(PIXIT_CMN_CMN_WakeUp - 5.0);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior--------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_021() runs on EVCC_Tester
                                                        system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
           v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
       } else {
           v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
       }
       preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                        v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_022() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
           v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
       } else {
           v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
       }
       preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                        v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_023() runs on EVCC_Tester
                                                       system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
       map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
       var float v_dataLinkprocessing;
       if(tc_CMN_ProcessingTime_Timer.running) {
           v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
       } else {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
        }
        preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                         v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
       preConVerdict :=
       f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
       log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions----------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_024() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-----------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
    if(preConVerdict == pass) {
       preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
    }

    if(preConVerdict == pass) {
        map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
        var float v_dataLinkprocessing;
        if(tc_CMN_ProcessingTime_Timer.running) {
            v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
        } else {
            v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
        }
        preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                         v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
    }

    if(preConVerdict == pass) {
       unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
       preConVerdict :=
       f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
    }

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
    }else {
       log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions----------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_025() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-----------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005();
   }else {
      log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_ChargeParameterDiscovery_026() runs on EVCC_Tester
                                                        system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_checkWakeUpConfig(sutInitiatedZeroPow);
   if(preConVerdict == pass) {
      preConVerdict := f_EVCC_AC_PR_SessionStop_002(v_HAL_61851_Listener);
   }

   if(preConVerdict == pass) {
      map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
      var float v_dataLinkprocessing;
      if(tc_CMN_ProcessingTime_Timer.running) {
          v_dataLinkprocessing := tc_CMN_ProcessingTime_Timer.read;
      } else {
          v_dataLinkprocessing := par_T_match_leave + par_CMN_Transmission_Delay;
      }
      preConVerdict := f_EVCC_startSleepingPhase(par_SECC_Pmax0W -
                          v_dataLinkprocessing - par_EVCC_WakeUpOffsetTime);
   }

   if(preConVerdict == pass) {
      unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, system:pt_V2G_TCP_TLS_ALM_EVCC_Port);
      preConVerdict :=
      f_EVCC_CMN_PR_AuthorizationPausedSession_001(v_HAL_61851_Listener);
   }

   //-------------- Test behavior-------------------------------------------------------
   if(preConVerdict==pass){
       f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004();
   }else {
      log("PreCondition was unsuccessful.");
   }


   //------------- Post Conditions------------------------------------------------------
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```

D.3.2.12 EVCC test cases for PowerDelivery

```
module TestCases_EVCC_AC_CMN_PowerDelivery {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_AC_VTB_PowerDelivery_001() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
       preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if( preConVerdict == pass ) {
           f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
       } else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_PowerDelivery_002() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
       preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if( preConVerdict == pass ) {
           f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED, v_HAL_61851_Listener);
       } else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_PowerDelivery_003() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
       preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if( preConVerdict == pass ) {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_SequenceError, v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_AC_VTB_PowerDelivery_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_UnknownSession, v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_AC_VTB_PowerDelivery_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_ChargingProfileInvalid, v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_AC_VTB_PowerDelivery_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_TariffSelectionInvalid, v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_AC_VTB_PowerDelivery_007() runs on EVCC_Tester system SystemEVCC {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_PowerDeliveryNotApplied, v_HAL_61851_Listener);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-----------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_AC_VTB_PowerDelivery_008() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_002(fAILED_ContactorError, v_HAL_61851_Listener);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-----------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_AC_VTB_PowerDelivery_009() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_003(v_HAL_61851_Listener);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-----------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

 testcase TC_EVCC_AC_VTB_PowerDelivery_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_004(v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_011() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_005(e_OpenCircuit);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_012() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_005(e_NegVolt12);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_013() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                    v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_014() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   // -------------- Pre Conditions-------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_015() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_016() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_017() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
   } else {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_018() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_019() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_020() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_021() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, fail);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_022() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_023() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_024() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
   } else {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_025() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_026() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_027() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
    } else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_028() runs on EVCC_Tester system SystemEVCC {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-----------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_029() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-----------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_030() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
   } else {
       log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-----------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_PowerDelivery_031() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //-------------- Test behavior-------------------------------------------------------
        if( preConVerdict == pass ) {
            f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
        } else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_PowerDelivery_032() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if( preConVerdict == pass ) {
            f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
        } else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_PowerDelivery_033() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(
                                                        v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if( preConVerdict == pass ) {
            f_EVCC_AC_TB_VTB_PowerDelivery_006(v_HAL_61851_Listener);
        } else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }
}
```

D.3.2.13 EVCC test cases for SessionStop

```
module TestCases_EVCC_DC_EIM_SessionStop {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_SessionStop all;
    import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from LibFunctions_15118_2 all;

testcase TC_EVCC_DC_VTB_SessionStop_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------------
    if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                pass, pass);
         }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_DC_VTB_SessionStop_004() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------------
     if(preConVerdict==pass){
           log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, pass);
         }
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-----------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

testcase TC_EVCC_DC_VTB_SessionStop_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------------
    if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                pass, pass);
         }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------------
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, pass);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                               pass, inconc);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_SessionStop_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, inconc);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

testcase TC_EVCC_DC_VTB_SessionStop_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                 pass, inconc);
         }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_DC_VTB_SessionStop_012() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, inconc);
         }
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

testcase TC_EVCC_DC_VTB_SessionStop_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                 inconc, pass);
         }
    }else {
        log("PreCondition was unsuccessful.");
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_DC_VTB_SessionStop_016() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
         }
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

testcase TC_EVCC_DC_VTB_SessionStop_017() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                 inconc, pass);
         }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_DC_VTB_SessionStop_018() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
         }
     }else {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

testcase TC_EVCC_DC_VTB_SessionStop_021() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                 inconc, pass);
         }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_DC_VTB_SessionStop_022() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
         }
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

testcase TC_EVCC_DC_VTB_SessionStop_023() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                 inconc, pass);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_DC_VTB_SessionStop_024() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             log("Send PowerDeliveryRes message with delay of 0.5s");
             sleep(0.5);
             if (getverdict == pass){
                 f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
             }
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

    testcase TC_EVCC_DC_VTB_SessionStop_027() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
             log("Send PowerDeliveryRes message with delay of 0.5s");
             sleep(0.5);
             if (getverdict == pass){
                 f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                    inconc, pass);
             }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_DC_VTB_SessionStop_028() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             log("Send PowerDeliveryRes message with delay of 0.5s");
             sleep(0.5);
             if (getverdict == pass){
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
                }
            }else {
                log("PreCondition was unsuccessful.");
            }

            //------------- Post Conditions-------------------------------------------------------
            f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
            f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
        }

        testcase TC_EVCC_DC_VTB_SessionStop_029() runs on EVCC_Tester system SystemEVCC {

            var HAL_61851_Listener v_HAL_61851_Listener;
            var verdicttype preConVerdict;

            // -------------- Pre Conditions-------------------------------------------------------
            f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
            preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

            //-------------- Test behavior--------------------------------------------------------
            if(preConVerdict==pass){
                log("Send PowerDeliveryRes message with delay of 0.5s");
                sleep(0.5);
                if (getverdict == pass){
                    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                        inconc, pass);
                }
            }else {
                log("PreCondition was unsuccessful.");
            }

            //------------- Post Conditions-------------------------------------------------------
            f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
            f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
        }

        testcase TC_EVCC_DC_VTB_SessionStop_030() runs on EVCC_Tester system SystemEVCC {

            var HAL_61851_Listener v_HAL_61851_Listener;
            var verdicttype preConVerdict;

            // -------------- Pre Conditions-------------------------------------------------------
            f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
            preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

            //-------------- Test behavior--------------------------------------------------------
            if(preConVerdict==pass){
                log("Send PowerDeliveryRes message with delay of 0.5s");
                sleep(0.5);
                if (getverdict == pass){
                    f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
                }
            }else {
                log("PreCondition was unsuccessful.");
            }


            //------------- Post Conditions-------------------------------------------------------
            f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
            f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
        }
}



module TestCases_EVCC_DC_CMN_SessionStop {

     import from DataStructure_HAL_61851 all;
     import from TestBehavior_EVCC_SessionStop all;
     import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
     import from Services_HAL_61851_1 all;
     import from TestBehavior_EVCC_CommonBehavior all;
     import from PreConditions_EVCC_15118_2 all;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  import from PostConditions_EVCC_15118_2 all;
  import from ComponentsAndPorts all;
  import from Configurations_15118_2 all;
  import from LibFunctions_15118_2 all;

testcase TC_EVCC_DC_VTB_SessionStop_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                  pass, pass);
         }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_DC_VTB_SessionStop_002() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, pass);
         }
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

  testcase TC_EVCC_DC_VTB_SessionStop_007() runs on EVCC_Tester system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
           log("Send PowerDeliveryRes message with delay of 0.5s");
           sleep(0.5);
           if (getverdict == pass){
               f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                    pass, inconc);
           }

      }else {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_DC_VTB_SessionStop_008() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             log("Send PowerDeliveryRes message with delay of 0.5s");
             sleep(0.5);
             if (getverdict == pass){
                 f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, inconc);
             }
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

    testcase TC_EVCC_DC_VTB_SessionStop_013() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
             log("Send PowerDeliveryRes message with delay of 0.5s");
             sleep(0.5);
             if (getverdict == pass){
                 f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                    inconc, pass);
             }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_DC_VTB_SessionStop_014() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             log("Send PowerDeliveryRes message with delay of 0.5s");
             sleep(0.5);
             if (getverdict == pass){
                 f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
             }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

testcase TC_EVCC_DC_VTB_SessionStop_019() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
                                                                inconc, pass);
         }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_DC_VTB_SessionStop_020() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
         }
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

testcase TC_EVCC_DC_VTB_SessionStop_025() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
         log("Send PowerDeliveryRes message with delay of 0.5s");
         sleep(0.5);
         if (getverdict == pass){
             f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(v_HAL_61851_Listener,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                                    inconc, pass);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_DC_VTB_SessionStop_026() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             log("Send PowerDeliveryRes message with delay of 0.5s");
             sleep(0.5);
             if (getverdict == pass){
                 f_EVCC_DC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass);
             }
         }else {
             log("PreCondition was unsuccessful.");
         }


         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}



module TestCases_EVCC_AC_EIM_SessionStop {

     import from DataStructure_HAL_61851 all;
     import from TestBehavior_EVCC_SessionStop all;
     import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
     import from Services_HAL_61851_1 all;
     import from TestBehavior_EVCC_CommonBehavior all;
     import from PreConditions_EVCC_15118_2 all;
     import from PostConditions_EVCC_15118_2 all;
     import from ComponentsAndPorts all;
     import from Configurations_15118_2 all;

    testcase TC_EVCC_AC_VTB_SessionStop_007() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass,
                                             pass, fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 testcase TC_EVCC_AC_VTB_SessionStop_008() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass,
                                          pass, fail);
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_AC_VTB_SessionStop_010() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass,
                                          inconc, fail);
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

  testcase TC_EVCC_AC_VTB_SessionStop_011() runs on EVCC_Tester system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if(preConVerdict==pass){
          f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass,
                                           inconc, fail);
      }else {
          log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_AC_VTB_SessionStop_013() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
       f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc,
                                        pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_AC_VTB_SessionStop_014() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-----------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

     //-------------- Test behavior-------------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc,
                                          pass, fail);
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-----------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_AC_VTB_SessionStop_016() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-----------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------------
    if(preConVerdict==pass){
       f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc,
                                        pass, fail);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-----------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_AC_VTB_SessionStop_017() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-----------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

     //-------------- Test behavior-------------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc,
                                          pass, fail);
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-----------------------------------------------------------
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_AC_VTB_SessionStop_019() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
            f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc,
                                             pass, fail);
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

      testcase TC_EVCC_AC_VTB_SessionStop_020() runs on EVCC_Tester system SystemEVCC {

          var HAL_61851_Listener v_HAL_61851_Listener;
          var verdicttype preConVerdict;

          // -------------- Pre Conditions-------------------------------------------------------
          f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
          preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

          //-------------- Test behavior--------------------------------------------------------
          if(preConVerdict==pass){
              f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc,
                                               pass, fail);
          }else {
              log("PreCondition was unsuccessful.");
          }


          //------------- Post Conditions-------------------------------------------------------
          f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
          f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_EVCC_AC_CMN_SessionStop {

     import from DataStructure_HAL_61851 all;
     import from TestBehavior_EVCC_SessionStop all;
     import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
     import from Services_HAL_61851_1 all;
     import from TestBehavior_EVCC_CommonBehavior all;
     import from PreConditions_EVCC_15118_2 all;
     import from PostConditions_EVCC_15118_2 all;
     import from ComponentsAndPorts all;
     import from Configurations_15118_2 all;

    testcase TC_EVCC_AC_VTB_SessionStop_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(preConVerdict==pass){
            f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass,
                                             pass, fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_AC_VTB_SessionStop_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_AC_TB_VTB_SessionStop_003(fAILED);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

testcase TC_EVCC_AC_VTB_SessionStop_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_003(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_AC_VTB_SessionStop_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_003(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
testcase TC_EVCC_AC_VTB_SessionStop_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_AC_TB_VTB_SessionStop_004();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_AC_VTB_SessionStop_006() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     preConVerdict := f_EVCC_AC_PR_SessionStop_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if(preConVerdict==pass){
         f_EVCC_AC_TB_VTB_SessionStop_005();
     }else {
         log("PreCondition was unsuccessful.");
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

   testcase TC_EVCC_AC_VTB_SessionStop_009() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict==pass){
           f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass,
                                            inconc, fail);
       }else {
           log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

   testcase TC_EVCC_AC_VTB_SessionStop_012() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if(preConVerdict==pass){
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc,
                                                pass, fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_AC_VTB_SessionStop_015() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc,
                                                pass, fail);
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_AC_VTB_SessionStop_018() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc,
                                                pass, fail);
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```

D.3.2.14 EVCC test cases for ChargingStatus

```
module TestCases_EVCC_AC_CMN_ChargingStatus {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargingStatus all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
testcase TC_EVCC_AC_VTB_ChargingStatus_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //-------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    //-------------- Test behavior --------------------------------------------------------
    if ( preConVerdict == pass ) {
      f_EVCC_AC_TB_VTB_ChargingStatus_001(v_HAL_61851_Listener, fail);
    }
    else {
      log ( "PreCondition was unsuccessful." );
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);

   }

 testcase TC_EVCC_AC_VTB_ChargingStatus_002() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     //-------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_ChargingStatus_002(fAILED, v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_AC_VTB_ChargingStatus_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //-------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if ( preConVerdict == pass ) {
      f_EVCC_AC_TB_VTB_ChargingStatus_002(fAILED_SequenceError, v_HAL_61851_Listener);
    }
    else {
      log ( "PreCondition was unsuccessful." );
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_AC_VTB_ChargingStatus_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //-------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //-------------- Test behavior-----------------------------------------------------------
    if ( preConVerdict == pass ) {
      f_EVCC_AC_TB_VTB_ChargingStatus_002(fAILED_UnknownSession, v_HAL_61851_Listener);
    }
    else {
      log ( "PreCondition was unsuccessful." );
    }

    //------------- Post Conditions----------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

 testcase TC_EVCC_AC_VTB_ChargingStatus_005() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     //-------------- Pre Conditions----------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

     //-------------- Test behavior-----------------------------------------------------------
     if ( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_ChargingStatus_003(v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     //------------- Post Conditions----------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

  testcase TC_EVCC_AC_VTB_ChargingStatus_006() runs on EVCC_Tester system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      //-------------- Pre Conditions----------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
      preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

      //-------------- Test behavior-----------------------------------------------------------
      if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingStatus_004(e_OpenCircuit);
      }
      else {
        log ( "PreCondition was unsuccessful." );
      }

      //------------- Post Conditions----------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
  }

  testcase TC_EVCC_AC_VTB_ChargingStatus_007() runs on EVCC_Tester system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      //-------------- Pre Conditions----------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
      preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

      //-------------- Test behavior-----------------------------------------------------------
      if ( preConVerdict == pass ) {
        f_EVCC_AC_TB_VTB_ChargingStatus_004(e_NegVolt12);
      }
      else {
        log ( "PreCondition was unsuccessful." );
      }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_ChargingStatus_008() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if ( preConVerdict == pass ) {
          f_EVCC_AC_TB_VTB_ChargingStatus_005(v_HAL_61851_Listener);
        }
        else {
          log ( "PreCondition was unsuccessful." );
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }
}
```

D.3.2.15 EVCC test cases for ChargingLoopAcOrPowerDelivery

```
module TestCases_EVCC_AC_PnC_ChargingLoopAcOrPowerDelivery {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargingLoopAcOrPowerDelivery all;
    import from Services_HAL_61851_1 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if ( preConVerdict == pass ) {
          f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_002(v_HAL_61851_Listener, fail);
        }
        else {
          log ( "PreCondition was unsuccessful." );
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_009() runs on EVCC_Tester
                                                                 system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if ( preConVerdict == pass ) {
          f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_006(v_HAL_61851_Listener, fail);
        }
        else {
          log ( "PreCondition was unsuccessful." );
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_014() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if ( preConVerdict == pass ) {
          f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_010(v_HAL_61851_Listener, fail);
        }
        else {
          log ( "PreCondition was unsuccessful." );
        }


        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }
}



module TestCases_EVCC_AC_CMN_ChargingLoopAcOrPowerDelivery {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargingLoopAcOrPowerDelivery all;
    import from Services_HAL_61851_1 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

  testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001() runs on EVCC_Tester
                                                              system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       // -------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
       preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

       // -------------- Test behavior--------------------------------------------------------
       if ( preConVerdict == pass ) {
         f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener, fail);
       }
       else {
         log ( "PreCondition was unsuccessful." );
       }

       // ------------- Post Conditions-------------------------------------------------------
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003() runs on EVCC_Tester
                                                            system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003(fAILED);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004() runs on EVCC_Tester
                                                            system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003(fAILED_SequenceError);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005() runs on EVCC_Tester
                                                            system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003(fAILED_UnknownSession);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006() runs on EVCC_Tester
                                                            system SystemEVCC {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if ( preConVerdict == pass ) {
      f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003(fAILED_ContactorError);
    }
    else {
      log ( "PreCondition was unsuccessful." );
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007() runs on EVCC_Tester
                                                            system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if ( preConVerdict == pass ) {
      f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_004();
    }
    else {
      log ( "PreCondition was unsuccessful." );
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

 testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_008() runs on EVCC_Tester
                                                             system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_005(v_HAL_61851_Listener, fail);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

 testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010() runs on EVCC_Tester
                                                             system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_007();
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011() runs on EVCC_Tester
                                                            system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_008(e_OpenCircuit);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012() runs on EVCC_Tester
                                                            system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_008(e_NegVolt12);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_013() runs on EVCC_Tester
                                                            system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

     // -------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
       f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_009(v_HAL_61851_Listener, fail);
     }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
          else {
            log ( "PreCondition was unsuccessful." );
          }


          // ------------- Post Conditions-------------------------------------------------------------
          f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
          f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
     }
}
```

D.3.2.16 EVCC test cases for MeteringReceipt

```
module TestCases_EVCC_DC_PnC_MeteringReceipt {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_MeteringReceipt all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

     testcase TC_EVCC_DC_VTB_MeteringReceipt_001() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
         preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

         // -------------- Test behavior--------------------------------------------------------------
         if (preConVerdict == pass) {
             f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
         }
         else {
              log("PreCondition was unsuccessful.");
         }

         // ------------- Post Conditions-------------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
     }

     testcase TC_EVCC_DC_VTB_MeteringReceipt_002() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         // -------------- Pre Conditions-------------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
         preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

         // -------------- Test behavior--------------------------------------------------------------
         if (preConVerdict == pass) {
             f_EVCC_DC_TB_VTB_MeteringReceipt_002(fAILED, v_HAL_61851_Listener);
         }
         else {
             log("PreCondition was unsuccessful.");
         }

         // ------------- Post Conditions-------------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
     }

     testcase TC_EVCC_DC_VTB_MeteringReceipt_003() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_002(fAILED_SequenceError, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_002(fAILED_UnknownSession, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_MeteringReceipt_002(fAILED_MeteringSignatureNotValid,
                                             v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_DC_TB_VTB_MeteringReceipt_002(fAILED_SignatureError, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_MeteringReceipt_001(v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_008() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_CMN_TB_VTB_MeteringReceipt_002(v_HAL_61851_Listener);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_MeteringReceipt_003(e_OpenCircuit);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_MeteringReceipt_003(e_NegVolt12);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // -------------- Test behavior-------------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // -------------- Test behavior-------------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // -------------- Test behavior-------------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_016() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // -------------- Test behavior-------------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_017() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // -------------- Test behavior---------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_018() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // -------------- Test behavior---------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_019() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    // -------------- Test behavior---------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_MeteringReceipt_020() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
           f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
        }
        else {
             log("PreCondition was unsuccessful.");
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_MeteringReceipt_021() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
           f_EVCC_DC_TB_VTB_MeteringReceipt_001(fail);
        }
        else {
             log("PreCondition was unsuccessful.");
        }


        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }
}



module TestCases_EVCC_AC_PnC_MeteringReceipt {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_MeteringReceipt all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_AC_VTB_MeteringReceipt_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
        }
        else {
             log("PreCondition was unsuccessful.");
        }

        // ------------- Post Conditions-------------------------------------------------------
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_002(fAILED, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_002(fAILED_SequenceError, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_002(fAILED_UnknownSession, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_002(fAILED_MeteringSignatureNotValid,
                                             v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // -------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

 testcase TC_EVCC_AC_VTB_MeteringReceipt_006() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if (preConVerdict == pass) {
         f_EVCC_AC_TB_VTB_MeteringReceipt_002(fAILED_SignatureError, v_HAL_61851_Listener);
     }
     else {
         log("PreCondition was unsuccessful.");
     }

     // -------------- Post Conditions------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_AC_VTB_MeteringReceipt_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_MeteringReceipt_001(v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // -------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

 testcase TC_EVCC_AC_VTB_MeteringReceipt_008() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if (preConVerdict == pass) {
         f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
     }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_009() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_010() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_CMN_TB_VTB_MeteringReceipt_002(v_HAL_61851_Listener);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_011() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_AC_TB_VTB_MeteringReceipt_003(e_OpenCircuit);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
testcase TC_EVCC_AC_VTB_MeteringReceipt_012() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_AC_TB_VTB_MeteringReceipt_003(e_NegVolt12);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_013() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
    f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_014() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_015() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // -------------- Test behavior-------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_016() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // -------------- Test behavior-------------------------------------------------------
    if (preConVerdict == pass) {
    f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_017() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // -------------- Test behavior-------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_AC_VTB_MeteringReceipt_018() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

    // -------------- Test behavior-------------------------------------------------------
    if (preConVerdict == pass) {
       f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
    }
    else {
         log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_MeteringReceipt_019() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
        f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
        }
        else {
             log("PreCondition was unsuccessful.");
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_MeteringReceipt_020() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
           f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
        }
        else {
             log("PreCondition was unsuccessful.");
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_AC_VTB_MeteringReceipt_021() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
           f_EVCC_AC_TB_VTB_MeteringReceipt_001(fail);
        }
        else {
             log("PreCondition was unsuccessful.");
        }


        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

D.3.2.17 EVCC test cases for CableCheck

```
module TestCases_EVCC_DC_CMN_CableCheck {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_V2GTPSDP all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PostConditions_EVCC_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from TestBehavior_EVCC_CableCheck all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_DC_VTB_CableCheck_001() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
          var verdicttype preConVerdict;

       //-------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if (preConVerdict == pass) {
            f_EVCC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, fail);
       }
       else{
            log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_CableCheck_002() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       //-------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_DC_PR_CableCheck_002(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if (preConVerdict == pass) {
            f_EVCC_DC_TB_VTB_CableCheck_002(fAILED, v_HAL_61851_Listener);
       }
       else{
            log("PreCondition was unsuccessful.");
       }

       //------------- Post Conditions-------------------------------------------------------
       f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
       f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_CableCheck_003() runs on EVCC_Tester system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;

       //-------------- Pre Conditions-------------------------------------------------------
       f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
       preConVerdict := f_EVCC_DC_PR_CableCheck_002(v_HAL_61851_Listener);

       //-------------- Test behavior--------------------------------------------------------
       if (preConVerdict == pass) {
            f_EVCC_DC_TB_VTB_CableCheck_002(fAILED_SequenceError, v_HAL_61851_Listener);
       }
       else{
            log("PreCondition was unsuccessful.");
       }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_CableCheck_004() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   //-------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_DC_PR_CableCheck_002(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_CableCheck_002(fAILED_UnknownSession, v_HAL_61851_Listener);
   }
   else{
        log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_CableCheck_005() runs on EVCC_Tester system SystemEVCC {

   var HAL_61851_Listener v_HAL_61851_Listener;
   var verdicttype preConVerdict;

   //-------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_DC_PR_CableCheck_002(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------
   if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_CableCheck_003(v_HAL_61851_Listener);
   }
   else{
        log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_DC_VTB_CableCheck_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //-------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_CableCheck_002(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
         f_EVCC_DC_TB_VTB_CableCheck_004(v_HAL_61851_Listener);
    }
    else{
         log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_DC_VTB_CableCheck_007() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_CableCheck_002(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
             f_EVCC_DC_TB_VTB_CableCheck_005(v_HAL_61851_Listener);
        }
        else{
             log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_EVCC_DC_VTB_CableCheck_008() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_CableCheck_002(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
             f_EVCC_DC_TB_VTB_CableCheck_006(e_OpenCircuit);
        }
        else{
             log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_EVCC_DC_VTB_CableCheck_009() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_CableCheck_002(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
             f_EVCC_DC_TB_VTB_CableCheck_006(e_NegVolt12);
        }
        else{
             log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }
}
```

D.3.2.18 EVCC test cases for PreCharge

```
module TestCases_EVCC_DC_CMN_PreCharge {

    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from TestBehavior_EVCC_V2GTPSDP all;
import from TestBehavior_EVCC_CommonBehavior all;
import from PostConditions_EVCC_15118_2 all;
import from PreConditions_EVCC_15118_2 all;
import from TestBehavior_EVCC_PreCharge all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;

testcase TC_EVCC_DC_VTB_PreCharge_001() runs on EVCC_Tester system SystemEVCC {

   var verdicttype preConVerdict;
   var HAL_61851_Listener v_HAL_61851_Listener;

   //-------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------

   if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_PreCharge_001(fail);
   }
   else{
        log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PreCharge_002() runs on EVCC_Tester system SystemEVCC {

   var verdicttype preConVerdict;
   var HAL_61851_Listener v_HAL_61851_Listener;

   //-------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------

   if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_PreCharge_002(fAILED, v_HAL_61851_Listener);
   }
   else{
        log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PreCharge_003() runs on EVCC_Tester system SystemEVCC {

   var verdicttype preConVerdict;
   var HAL_61851_Listener v_HAL_61851_Listener;

   //-------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------

   if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_PreCharge_002(fAILED_SequenceError, v_HAL_61851_Listener);
   }
   else{
        log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}

testcase TC_EVCC_DC_VTB_PreCharge_004() runs on EVCC_Tester system SystemEVCC {

   var verdicttype preConVerdict;
   var HAL_61851_Listener v_HAL_61851_Listener;

   //-------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------

   if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_PreCharge_002(fAILED_UnknownSession, v_HAL_61851_Listener);
   }
   else{
        log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_PreCharge_005() runs on EVCC_Tester system SystemEVCC {

   var verdicttype preConVerdict;
   var HAL_61851_Listener v_HAL_61851_Listener;

   //-------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
   preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

   //-------------- Test behavior--------------------------------------------------------

   if (preConVerdict == pass) {
       f_EVCC_DC_TB_VTB_PreCharge_003(v_HAL_61851_Listener);
   }
   else{
        log("PreCondition was unsuccessful.");
   }

   //------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_DC_VTB_PreCharge_006() runs on EVCC_Tester system SystemEVCC {

    var verdicttype preConVerdict;
    var HAL_61851_Listener v_HAL_61851_Listener;

    //-------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------

    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_PreCharge_004(v_HAL_61851_Listener);
    }
    else{
         log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
 }

 testcase TC_EVCC_DC_VTB_PreCharge_007() runs on EVCC_Tester system SystemEVCC {

    var verdicttype preConVerdict;
    var HAL_61851_Listener v_HAL_61851_Listener;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      //-------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------

      if (preConVerdict == pass) {
          f_EVCC_DC_TB_VTB_PreCharge_005(v_HAL_61851_Listener);
      }
      else{
           log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

   testcase TC_EVCC_DC_VTB_PreCharge_008() runs on EVCC_Tester system SystemEVCC {

      var verdicttype preConVerdict;
      var HAL_61851_Listener v_HAL_61851_Listener;

      //-------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------

      if (preConVerdict == pass) {
          f_EVCC_DC_TB_VTB_PreCharge_006(e_OpenCircuit);
      }
      else{
           log("PreCondition was unsuccessful.");
      }

      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }

   testcase TC_EVCC_DC_VTB_PreCharge_009() runs on EVCC_Tester system SystemEVCC {

      var verdicttype preConVerdict;
      var HAL_61851_Listener v_HAL_61851_Listener;

      //-------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
      preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------

      if (preConVerdict == pass) {
          f_EVCC_DC_TB_VTB_PreCharge_006(e_NegVolt12);
      }
      else{
           log("PreCondition was unsuccessful.");
      }


      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
   }
}
```

D.3.2.19 EVCC test cases for PreChargeOrPowerDelivery

```
module TestCases_EVCC_DC_CMN_PreChargeOrPowerDelivery {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  import from Pics_15118_2 all;
  import from DataStructure_HAL_61851 all;
  import from TestBehavior_EVCC_PreChargeOrPowerDelivery all;
  import from Services_HAL_61851_1 all;
  import from PreConditions_EVCC_15118_2 all;
  import from PostConditions_EVCC_15118_2 all;
  import from ComponentsAndPorts all;
  import from Configurations_15118_2 all;

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_001() runs on EVCC_Tester
                                                         system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //-------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if ( preConVerdict == pass ) {
         f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001(fail);
    }
    else {
      log ( "PreCondition was unsuccessful." );
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

  testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002() runs on EVCC_Tester
                                                         system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
      preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

      // -------------- Test behavior--------------------------------------------------------
      if ( preConVerdict == pass ) {
           f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED, v_HAL_61851_Listener);
      }
      else {
        log ( "PreCondition was unsuccessful." );
      }

      // ------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
  }

  testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003() runs on EVCC_Tester
                                                         system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      // -------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
      preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

      // -------------- Test behavior--------------------------------------------------------
      if ( preConVerdict == pass ) {
           f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_SequenceError,
                                                            v_HAL_61851_Listener);
      }
      else {
        log ( "PreCondition was unsuccessful." );
      }

      // ------------- Post Conditions-------------------------------------------------------
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_UnknownSession,
                                                          v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-----------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype        preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_ChargingProfileInvalid,
                                                          v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-----------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_TariffSelectionInvalid,
                                                          v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-----------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_PowerDeliveryNotApplied,
                                                          v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(fAILED_ContactorError,
                                                          v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_003(v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

 testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010() runs on EVCC_Tester
                                                         system SystemEVCC {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_004(v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_005(e_OpenCircuit);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_005(e_NegVolt12);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     //-------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     //-------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     //------------- Post Conditions------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     //-------------- Pre Conditions------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

     //-------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     //------------- Post Conditions------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_015() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     //-------------- Pre Conditions------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

     //-------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001(fail);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     //------------- Post Conditions------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_016() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     //-------------- Pre Conditions------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

     //-------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001(fail);
     }
     else {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        log ( "PreCondition was unsuccessful." );
      }

      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
  }

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017() runs on EVCC_Tester
                                                        system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      //-------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
      preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if ( preConVerdict == pass ) {
           f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
      }
      else {
        log ( "PreCondition was unsuccessful." );
      }

      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
  }

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018() runs on EVCC_Tester
                                                        system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      //-------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
      preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if ( preConVerdict == pass ) {
           f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
      }
      else {
        log ( "PreCondition was unsuccessful." );
      }

      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019() runs on EVCC_Tester
                                                        system SystemEVCC {

      var HAL_61851_Listener v_HAL_61851_Listener;
      var verdicttype preConVerdict;

      //-------------- Pre Conditions-------------------------------------------------------
      f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
      preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

      //-------------- Test behavior--------------------------------------------------------
      if ( preConVerdict == pass ) {
           f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
      }
      else {
        log ( "PreCondition was unsuccessful." );
      }

      //------------- Post Conditions-------------------------------------------------------
      f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
      f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 }

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     //-------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     //-------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
 }

testcase TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022() runs on EVCC_Tester
                                                        system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     //-------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_PreChargeRenegotiation_001(v_HAL_61851_Listener);

     //-------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
          f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006();
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }


     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

D.3.2.20 EVCC test cases for CurrentDemand

```
module TestCases_EVCC_DC_CMN_CurrentDemand {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from TestBehavior_EVCC_CurrentDemand all;
    import from TestBehavior_EVCC_MeteringReceipt all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2  all;
    import from PostConditions_EVCC_15118_2  all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

    testcase TC_EVCC_DC_VTB_CurrentDemand_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
           var verdicttype preConVerdict;

           //-------------- Pre Conditions-------------------------------------------------------
           f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
           preConVerdict := f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);

           //-------------- Test behavior --------------------------------------------------------
        if ( preConVerdict == pass ) {
            f_EVCC_DC_TB_VTB_CurrentDemand_001(fail);
           }
           else {
             log("PreCondition was unsuccessful.");
           }

           //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
           f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

     testcase TC_EVCC_DC_VTB_CurrentDemand_002() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         //-------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
         preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

         //-------------- Test behavior --------------------------------------------------------
         if ( preConVerdict == pass ) {
             f_EVCC_DC_TB_VTB_CurrentDemand_002(fAILED, v_HAL_61851_Listener);
         }
         else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
     }

     testcase TC_EVCC_DC_VTB_CurrentDemand_003() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;

         //-------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
         preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

         //-------------- Test behavior --------------------------------------------------------
         if ( preConVerdict == pass ) {
             f_EVCC_DC_TB_VTB_CurrentDemand_002(fAILED_SequenceError, v_HAL_61851_Listener);
         }
         else {
             log("PreCondition was unsuccessful.");
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_CurrentDemand_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //-------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //-------------- Test behavior --------------------------------------------------------
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_CurrentDemand_002(fAILED_UnknownSession, v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_CurrentDemand_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //-------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //-------------- Test behavior --------------------------------------------------------
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_CurrentDemand_003(v_HAL_61851_Listener);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_CurrentDemand_006() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    //-------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //-------------- Test behavior --------------------------------------------------------
    if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_CurrentDemand_004(e_OpenCircuit);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_CurrentDemand_007() runs on EVCC_Tester system SystemEVCC {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        //-------------- Test behavior --------------------------------------------------------
        if ( preConVerdict == pass ) {
            f_EVCC_DC_TB_VTB_CurrentDemand_004(e_NegVolt12);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_CurrentDemand_008() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        //-------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        //-------------- Test behavior --------------------------------------------------------
        if ( preConVerdict == pass ) {
            f_EVCC_DC_TB_VTB_CurrentDemand_005(v_HAL_61851_Listener);
        }
        else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }
}
```

D.3.2.21 EVCC test cases for ChargingLoopDcOrPowerDelivery

```
module TestCases_EVCC_DC_PnC_ChargingLoopDcOrPowerDelivery {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargingLoopDcOrPowerDelivery all;
    import from Services_HAL_61851_1 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

     testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002() runs on EVCC_Tester
                                                                 system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if ( preConVerdict == pass ) {
           f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_002(v_HAL_61851_Listener, fail);
        }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        else {
          log ( "PreCondition was unsuccessful." );
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }
       testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_009() runs on EVCC_Tester
                                                                    system SystemEVCC {

          var HAL_61851_Listener v_HAL_61851_Listener;
          var verdicttype preConVerdict;

          // -------------- Pre Conditions-------------------------------------------------------
          f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
          preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

          // -------------- Test behavior--------------------------------------------------------
          if ( preConVerdict == pass ) {
           f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_006(fail);
          }
          else {
            log ( "PreCondition was unsuccessful." );
          }

          // ------------- Post Conditions-------------------------------------------------------
          f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
          f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
      }

       testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_014() runs on EVCC_Tester
                                                                    system SystemEVCC {

          var HAL_61851_Listener v_HAL_61851_Listener;
          var verdicttype preConVerdict;

          // -------------- Pre Conditions-------------------------------------------------------
          f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
          preConVerdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);

          // -------------- Test behavior--------------------------------------------------------
          if ( preConVerdict == pass ) {
           f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_010(fail);
          }
          else {
            log ( "PreCondition was unsuccessful." );
          }


          // ------------- Post Conditions-------------------------------------------------------
          f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
          f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
      }
}



module TestCases_EVCC_DC_CMN_ChargingLoopDcOrPowerDelivery {

    import from Pics_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_ChargingLoopDcOrPowerDelivery all;
    import from Services_HAL_61851_1 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;

   testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001() runs on EVCC_Tester
                                                                    system SystemEVCC {

       var HAL_61851_Listener v_HAL_61851_Listener;
       var verdicttype preConVerdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   // -------------- Pre Conditions-------------------------------------------------------
   f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
   preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

   // -------------- Test behavior--------------------------------------------------------
   if ( preConVerdict == pass ) {
      f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener, fail);
   }
   else {
     log ( "PreCondition was unsuccessful." );
   }

   // ------------- Post Conditions-------------------------------------------------------
   f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
   f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003() runs on EVCC_Tester
                                                            system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003(fAILED, v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004() runs on EVCC_Tester
                                                            system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior--------------------------------------------------------
     if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003(fAILED_SequenceError,
                                                           v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005() runs on EVCC_Tester
                                                            system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     // -------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003(fAILED_UnknownSession,
                                                           v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-----------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006() runs on EVCC_Tester
                                                            system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003(fAILED_ContactorError,
                                                           v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-----------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007() runs on EVCC_Tester
                                                            system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

     // -------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_004(v_HAL_61851_Listener);
     }
     else {
       log ( "PreCondition was unsuccessful." );
     }

     // ------------- Post Conditions-----------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

 testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_008() runs on EVCC_Tester
                                                             system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;

     // -------------- Pre Conditions------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
     preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

     // -------------- Test behavior-------------------------------------------------------
     if ( preConVerdict == pass ) {
        f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_005(fail);
     }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        else {
          log ( "PreCondition was unsuccessful." );
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
      }

    testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if ( preConVerdict == pass ) {
         f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_007(v_HAL_61851_Listener);
        }
        else {
          log ( "PreCondition was unsuccessful." );
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if ( preConVerdict == pass ) {
         f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_008(e_OpenCircuit);
        }
        else {
          log ( "PreCondition was unsuccessful." );
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if ( preConVerdict == pass ) {
         f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_008(e_NegVolt12);
        }
        else {
          log ( "PreCondition was unsuccessful." );
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

     testcase TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_013() runs on EVCC_Tester
                                                                 system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if ( preConVerdict == pass ) {
           f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_009(fail);
        }
        else {
          log ( "PreCondition was unsuccessful." );
        }


        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
      }
}
```

D.3.2.22 EVCC test cases for WeldingDetection

```
module TestCases_EVCC_DC_CMN_WeldingDetection {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_WeldingDetection all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118_2 all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from LibFunctions_15118_2 all;

    testcase TC_EVCC_DC_VTB_WeldingDetection_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
            log("Send PowerDeliveryRes message with delay of 0.5s");
            sleep(0.5);
            if (getverdict == pass){
                f_EVCC_DC_TB_VTB_WeldingDetection_001(v_HAL_61851_Listener, fail);
            }
        }
        else {
             log("PreCondition was unsuccessful.");
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_WeldingDetection_002() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_WeldingDetection_002(fAILED);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_WeldingDetection_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_WeldingDetection_002(fAILED_SequenceError);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_WeldingDetection_004() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_WeldingDetection_002(fAILED_UnknownSession);
    }
    else {
        log("PreCondition was unsuccessful.");
    }

    // ------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
}

testcase TC_EVCC_DC_VTB_WeldingDetection_005() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

    // -------------- Test behavior--------------------------------------------------------
    if (preConVerdict == pass) {
        f_EVCC_DC_TB_VTB_WeldingDetection_003();
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_WeldingDetection_006() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

        // -------------- Test behavior---------------------------------------------------------
        if (preConVerdict == pass) {
            f_EVCC_DC_TB_VTB_WeldingDetection_004();
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_WeldingDetection_007() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

        // -------------- Test behavior---------------------------------------------------------
        if (preConVerdict == pass) {
            f_EVCC_DC_TB_VTB_WeldingDetection_005(e_OpenCircuit);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }

    testcase TC_EVCC_DC_VTB_WeldingDetection_008() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);

        // -------------- Test behavior---------------------------------------------------------
        if (preConVerdict == pass) {
            f_EVCC_DC_TB_VTB_WeldingDetection_005(e_NegVolt12);
        }
        else {
            log("PreCondition was unsuccessful.");
        }

        // ------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    testcase TC_EVCC_DC_VTB_WeldingDetection_009() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener, system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        // -------------- Test behavior--------------------------------------------------------
        if (preConVerdict == pass) {
             log("Send PowerDeliveryRes message with delay of 0.5s");
            sleep(0.5);
            if (getverdict == pass){
                f_EVCC_DC_TB_VTB_WeldingDetection_006(v_HAL_61851_Listener);
            }
        }
        else {
             log("PreCondition was unsuccessful.");
        }


        // ------------- Post Conditions-----------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener, system);
    }
}
```

D.3.2.23 EVCC test cases for WeldingDetectionOrSessionStop

```
module TestCases_EVCC_DC_EIM_WeldingDetectionOrSessionStop {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from LibFunctions_15118_2 all;

     testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_007() runs on EVCC_Tester
                                                                 system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            log("Send PowerDeliveryRes message with delay of 0.5s");
            sleep(0.5);
            if (getverdict == pass){
                f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                   pass, pass, fail);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-----------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_008() runs on EVCC_Tester
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                pass, pass, fail);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_010() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                pass, inconc, fail);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_011() runs on EVCC_Tester
                                                        system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                pass, inconc, fail);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    //------------- Post Conditions-------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013() runs on EVCC_Tester
                                                            system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                               inconc, pass, fail);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014() runs on EVCC_Tester
                                                            system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                               inconc, pass, fail);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016() runs on EVCC_Tester
                                                            system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                                    inconc, pass, fail);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017() runs on EVCC_Tester
                                                            system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                    inconc, pass, fail);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019() runs on EVCC_Tester
                                                            system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                    inconc, pass, fail);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020() runs on EVCC_Tester
                                                            system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            log("Send PowerDeliveryRes message with delay of 0.5s");
            sleep(0.5);
            if (getverdict == pass){
                f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                   inconc, pass, fail);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-----------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}



module TestCases_EVCC_DC_CMN_WeldingDetectionOrSessionStop {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from LibFunctions_15118_2 all;

     testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_001() runs on EVCC_Tester
                                                                 system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-----------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            log("Send PowerDeliveryRes message with delay of 0.5s");
            sleep(0.5);
            if (getverdict == pass){
                f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                   pass, pass, fail);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-----------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-----------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004(fAILED);
        }else {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003() runs on EVCC_Tester
                                                            system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004(fAILED_SequenceError);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004() runs on EVCC_Tester
                                                            system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004(fAILED_UnknownSession);
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005() runs on EVCC_Tester
                                                            system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions--------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener);

    //-------------- Test behavior---------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_005();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006() runs on EVCC_Tester
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_006();
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_009() runs on EVCC_Tester
                                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                  pass, inconc, fail);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012() runs on EVCC_Tester
                                                                    system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

    //-------------- Test behavior--------------------------------------------------------
    if(preConVerdict==pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                  inconc, pass, fail);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            log("Send PowerDeliveryRes message with delay of 0.5s");
            sleep(0.5);
            if (getverdict == pass){
                f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                   inconc, pass, fail);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }

        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018() runs on EVCC_Tester
                                                                system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);

        //-------------- Test behavior--------------------------------------------------------
        if(preConVerdict==pass){
            log("Send PowerDeliveryRes message with delay of 0.5s");
            sleep(0.5);
            if (getverdict == pass){
                f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                   inconc, pass, fail);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Annex E
(normative)

Test-case specifications for 15118-2 Reliability and Robustness Testing

E.1 General information

This annex covers all Reliability and Robustness test case *specifications* in the ATS.

E.2 SECC test cases

```
module TestCases_SECC_DC_CMN_Reliability {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_Reliability all;
    import from TestBehavior_SECC_PowerDelivery all;
    import from TestBehavior_SECC_WeldingDetection all;
    import from TestBehavior_SECC_SessionStop all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from ComponentsAndPorts all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from Configurations_15118_2 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Timer_15118_2 all;

     testcase TC_SECC_DC_VTB_Reliability_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
        var float v_timeMin := 2.0;
        var integer v_numberOfLoops := 20;
        vc_unmapSlac := false;

        //-------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
            preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

            //-------------- Test behavior-------------------------------------------------------
            if(preConVerdict==pass){
                f_SECC_DC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
                if(getverdict == pass) {
                    vc_DC_EVStatus.eVReady := false;
                    f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                                       true, true, inconc);
                }
                if(getverdict == pass) {
                    f_SECC_DC_TB_VTB_WeldingDetection_001(inconc);
                }
                if(getverdict == pass) {
                    f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener,
                                                      inconc);
                }
            }else {
                log("PreCondition was unsuccessful.");
            }
            if(getverdict != pass) {
                break;
            } else {
                if(not(i == (v_numberOfLoops-1))) {
                    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
                    pt_HAL_61851_Internal_Port.clear;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                v_HAL_61851_Listener.start(f_SECC_HAL_61851_Listener(false, false, e_StateA));
                vc_ChargeProgress := start_;
            }
        }
    }
    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_Reliability_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
    var float v_timeMin := 15.0;
    var integer v_numberOfLoops := 2;
    vc_unmapSlac := false;

    //-------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
        preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_DC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
            if(getverdict == pass) {
                vc_DC_EVStatus.eVReady := false;
                f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                                   true, true, inconc);
            }
            if(getverdict == pass) {
                f_SECC_DC_TB_VTB_WeldingDetection_001(inconc);
            }
            if(getverdict == pass) {
                f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener,
                                                  inconc);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }
        if(getverdict != pass) {
            break;
        } else {
            if(not(i == (v_numberOfLoops-1))) {
                f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
                pt_HAL_61851_Internal_Port.clear;
                v_HAL_61851_Listener.start(f_SECC_HAL_61851_Listener(false, false, e_StateA));
                vc_ChargeProgress := start_;
            }
        }
    }
    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_DC_VTB_Reliability_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
    var float v_timeMin := 45.0;

    //-------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_DC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
        if(getverdict == pass) {
            vc_DC_EVStatus.eVReady := false;
            f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                               true, true, inconc);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            if(getverdict == pass) {
                f_SECC_DC_TB_VTB_WeldingDetection_001(inconc);
            }
            if(getverdict == pass) {
                f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener,
                                                  inconc);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }


}



module TestCases_SECC_AC_CMN_Reliability {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_Reliability all;
    import from TestBehavior_SECC_PowerDelivery all;
    import from TestBehavior_SECC_SessionStop all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from ComponentsAndPorts all;
    import from PreConditions_SECC_15118_2 all;
    import from PostConditions_SECC_15118_2 all;
    import from Configurations_15118_2 all;
    import from Timer_15118 all;
    import from Timer_15118_2 all;
    import from Pics_15118 all;

     testcase TC_SECC_AC_VTB_Reliability_001() runs on SECC_Tester system SystemSECC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
        var float v_timeMin := 2.0;
        var integer v_numberOfLoops := 20;
        vc_unmapSlac := false;

        //-------------- Pre Conditions-------------------------------------------------------
        f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
            preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

            //-------------- Test behavior--------------------------------------------------------
            if(preConVerdict==pass){
                f_SECC_AC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
                if(getverdict == pass) {
                    f_SECC_AC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                                       inconc);
                }
                if(getverdict == pass) {
                    f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener,
                                                      inconc);
                }
            }else {
                log("PreCondition was unsuccessful.");
            }
            if(getverdict != pass) {
                break;
            } else {
                if(not(i == (v_numberOfLoops-1))) {
                    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                    unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
                    pt_HAL_61851_Internal_Port.clear;
                    v_HAL_61851_Listener.start(f_SECC_HAL_61851_Listener(false, false, e_StateA));
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                vc_ChargeProgress := start_;
            }
        }
    }
    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_Reliability_002() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
    var float v_timeMin := 15.0;
    var integer v_numberOfLoops := 2;
    vc_unmapSlac := false;

    //-------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
        preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_SECC_AC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
            if(getverdict == pass) {
                f_SECC_AC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                                   inconc);
            }
            if(getverdict == pass) {
                f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener,
                                                  inconc);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }
        if(getverdict != pass) {
            break;
        } else {
            if(not(i == (v_numberOfLoops-1))) {
                f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, system:pt_V2G_TCP_TLS_ALM_SECC_Port);
                pt_HAL_61851_Internal_Port.clear;
                v_HAL_61851_Listener.start(f_SECC_HAL_61851_Listener(false, false, e_StateA));
                vc_ChargeProgress := start_;
            }
        }
    }
    //------------- Post Conditions-------------------------------------------------------
    f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_SECC_AC_VTB_Reliability_003() runs on SECC_Tester system SystemSECC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
    var float v_timeMin := 45.0;

    //-------------- Pre Conditions-------------------------------------------------------
    f_SECC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_SECC_AC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
        if(getverdict == pass) {
            f_SECC_AC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                               inconc);
        }
        if(getverdict == pass) {
            f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener,
                                              inconc);
        }
    }else {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            log("PreCondition was unsuccessful.");
        }


        //------------- Post Conditions-------------------------------------------------------
        f_SECC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_SECC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }


}
```

E.3 EVCC test cases

```
module TestCases_EVCC_DC_PnC_Reliability {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_Reliability all;
    import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118_2 all;
    import from Timer_15118 all;
    import from Pics_15118 all;

    testcase TC_EVCC_DC_VTB_Reliability_005() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
        var float v_timeMin := 2.0;
        var integer v_numberOfLoops := 20;
        vc_unmapSlac := false;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
            preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

            //-------------- Test behavior-------------------------------------------------------
            if(preConVerdict==pass){
                f_EVCC_DC_TB_VTB_Reliability_002(v_HAL_61851_Listener, v_timeMin);
                log("Send PowerDeliveryRes message with delay of 0.5s");
                sleep(0.5);
                if (getverdict == pass){
                    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                        pass, pass, inconc);
                }
            }else {
                log("PreCondition was unsuccessful.");
            }
            if(getverdict != pass) {
                break;
             } else {
                 if(not(i == (v_numberOfLoops-1))) {
                     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                     pt_HAL_61851_Internal_Port.clear;
                     pt_V2G_TCP_TLS_Port_Control.clear;
                     v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(false));
                     vc_V2G_Port_IpAddress := par_SECC_IPaddress;
                     vc_V2G_Port_PortNumber := 0;
                     vc_DC_EVSEStatus := {notificationMaxDelay := par_EVCC_NotificationMaxDelay,
                                          eVSENotification := none_,
                                          eVSEIsolationStatus := invalid,
                                          eVSEStatusCode := eVSE_Ready};
                 }
             }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_Reliability_006() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
        var float v_timeMin := 15.0;
        var integer v_numberOfLoops := 2;
        vc_unmapSlac := false;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
            preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

            //-------------- Test behavior-------------------------------------------------------------
            if(preConVerdict==pass){
                f_EVCC_DC_TB_VTB_Reliability_002(v_HAL_61851_Listener, v_timeMin);
                log("Send PowerDeliveryRes message with delay of 0.5s");
                sleep(0.5);
                if (getverdict == pass){
                    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                        pass, pass, inconc);
                }
            }else {
                log("PreCondition was unsuccessful.");
            }
            if(getverdict != pass) {
                break;
             } else {
                 if(not(i == (v_numberOfLoops-1))) {
                     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                     pt_HAL_61851_Internal_Port.clear;
                     pt_V2G_TCP_TLS_Port_Control.clear;
                     v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(false));
                     vc_V2G_Port_IpAddress := par_SECC_IPaddress;
                     vc_V2G_Port_PortNumber := 0;
                     vc_DC_EVSEStatus := {notificationMaxDelay := par_EVCC_NotificationMaxDelay,
                                          eVSENotification := none_,
                                          eVSEIsolationStatus := invalid,
                                          eVSEStatusCode := eVSE_Ready};
                 }
             }
         }
         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

    testcase TC_EVCC_DC_VTB_Reliability_007() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
        var float v_timeMin := 45.0;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_DC_TB_VTB_Reliability_002(v_HAL_61851_Listener, v_timeMin);
            log("Send PowerDeliveryRes message with delay of 0.5s");
            sleep(0.5);
            if (getverdict == pass){
                f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                    pass, pass, inconc);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        //------------- Post Conditions-------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }


}



module TestCases_EVCC_DC_CMN_Reliability {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_Reliability all;
    import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118_2 all;
    import from Timer_15118 all;
    import from Pics_15118 all;

    testcase TC_EVCC_DC_VTB_Reliability_001() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
        var float v_timeMin := 2.0;
        var integer v_numberOfLoops := 20;
        vc_unmapSlac := false;

        // -------------- Pre Conditions-------------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
            preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

            //-------------- Test behavior-------------------------------------------------------------
            if(preConVerdict==pass){
                f_EVCC_DC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
                log("Send PowerDeliveryRes message with delay of 0.5s");
                sleep(0.5);
                if (getverdict == pass){
                    f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                        pass, pass, inconc);
                }
            }else {
                log("PreCondition was unsuccessful.");
            }
            if(getverdict != pass) {
                break;
             } else {
                 if(not(i == (v_numberOfLoops-1))) {
                     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                     pt_HAL_61851_Internal_Port.clear;
                     pt_V2G_TCP_TLS_Port_Control.clear;
                     v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(false));
                     vc_V2G_Port_IpAddress := par_SECC_IPaddress;
                     vc_V2G_Port_PortNumber := 0;
                     vc_DC_EVSEStatus := {notificationMaxDelay := par_EVCC_NotificationMaxDelay,
                                          eVSENotification := none_,
                                         eVSEIsolationStatus := invalid,
                                         eVSEStatusCode := eVSE_Ready};
                 }
             }
        }
        //------------- Post Conditions-------------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
testcase TC_EVCC_DC_VTB_Reliability_002() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
    var float v_timeMin := 15.0;
    var integer v_numberOfLoops := 2;
    vc_unmapSlac := false;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
        preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_DC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
            log("Send PowerDeliveryRes message with delay of 0.5s");
            sleep(0.5);
            if (getverdict == pass){
                f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                    pass, pass, inconc);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }
        if(getverdict != pass) {
            break;
         } else {
             if(not(i == (v_numberOfLoops-1))) {
                 f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                 pt_HAL_61851_Internal_Port.clear;
                 pt_V2G_TCP_TLS_Port_Control.clear;
                 v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(false));
                 vc_V2G_Port_IpAddress := par_SECC_IPaddress;
                 vc_V2G_Port_PortNumber := 0;
                 vc_DC_EVSEStatus := {notificationMaxDelay := par_EVCC_NotificationMaxDelay,
                                      eVSENotification := none_,
                                      eVSEIsolationStatus := invalid,
                                      eVSEStatusCode := eVSE_Ready};
             }
         }
     }
     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

testcase TC_EVCC_DC_VTB_Reliability_003() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
    var float v_timeMin := 45.0;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    preConVerdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    //-------------- Test behavior-------------------------------------------------------
    if(preConVerdict==pass){
        f_EVCC_DC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                pass, pass, inconc);
        }
    }else {
        log("PreCondition was unsuccessful.");
    }

    //------------- Post Conditions-------------------------------------------------------
    f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
    f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}


module TestCases_EVCC_AC_PnC_Reliability {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_Reliability all;
    import from TestBehavior_EVCC_SessionStop all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
    import from PostConditions_EVCC_15118_2 all;
    import from ComponentsAndPorts all;
    import from Configurations_15118_2 all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118_2 all;
    import from Timer_15118 all;
    import from Pics_15118 all;

    testcase TC_EVCC_AC_VTB_Reliability_005() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
        var float v_timeMin := 2.0;
        var integer v_numberOfLoops := 20;
        vc_unmapSlac := false;

        // -------------- Pre Conditions-------------------------------------------------------
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
            preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

            //-------------- Test behavior-------------------------------------------------------
            if(preConVerdict==pass){
                f_EVCC_AC_TB_VTB_Reliability_002(v_HAL_61851_Listener, v_timeMin);
                if (getverdict == pass){
                    f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, pass,
                                                     inconc);
                }
            }else {
                log("PreCondition was unsuccessful.");
            }
            if(getverdict != pass) {
                break;
             } else {
                 if(not(i == (v_numberOfLoops-1))) {
                     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                     pt_HAL_61851_Internal_Port.clear;
                     pt_V2G_TCP_TLS_Port_Control.clear;
                     v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(false));
                     vc_V2G_Port_IpAddress := par_SECC_IPaddress;
                     vc_V2G_Port_PortNumber := 0;
                     vc_AC_EVSEStatus := { rCD := false,
                                           notificationMaxDelay := par_EVCC_NotificationMaxDelay,
                                           eVSENotification := none_ };
                 }
             }
        }
        //------------- Post Conditions-------------------------------------------------------
        f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
        f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
    }

     testcase TC_EVCC_AC_VTB_Reliability_006() runs on EVCC_Tester system SystemEVCC {

        var HAL_61851_Listener v_HAL_61851_Listener;
        var verdicttype preConVerdict;
        var float v_timeMin := 15.0;
        var integer v_numberOfLoops := 2;
        vc_unmapSlac := false;

        // -------------- Pre Conditions-------------------------------------------------------
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
        for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
            preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

            //-------------- Test behavior-------------------------------------------------------
            if(preConVerdict==pass){
               f_EVCC_AC_TB_VTB_Reliability_002(v_HAL_61851_Listener, v_timeMin);
               if (getverdict == pass){
                   f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, pass,
                                                    inconc);
               }
            }else {
               log("PreCondition was unsuccessful.");
            }
            if(getverdict != pass) {
                break;
             } else {
                 if(not(i == (v_numberOfLoops-1))) {
                     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                     pt_HAL_61851_Internal_Port.clear;
                     pt_V2G_TCP_TLS_Port_Control.clear;
                     v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(false));
                     vc_V2G_Port_IpAddress := par_SECC_IPaddress;
                     vc_V2G_Port_PortNumber := 0;
                     vc_AC_EVSEStatus := { rCD := false,
                                           notificationMaxDelay := par_EVCC_NotificationMaxDelay,
                                           eVSENotification := none_ };
                 }
             }
         }
         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_EVCC_AC_VTB_Reliability_007() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;
         var float v_timeMin := 45.0;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

            //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_AC_TB_VTB_Reliability_002(v_HAL_61851_Listener, v_timeMin);
             if (getverdict == pass){
                 f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, pass,
                                                  inconc);
             }
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }


}



module TestCases_EVCC_AC_CMN_Reliability {

    import from DataStructure_HAL_61851 all;
    import from TestBehavior_EVCC_SessionStop all;
    import from TestBehavior_EVCC_Reliability all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from PreConditions_EVCC_15118_2 all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from PostConditions_EVCC_15118_2 all;
import from ComponentsAndPorts all;
import from Configurations_15118_2 all;
import from LibFunctions_15118_2 all;
import from Timer_15118_2 all;
import from Timer_15118 all;
import from Pics_15118 all;

testcase TC_EVCC_AC_VTB_Reliability_001() runs on EVCC_Tester system SystemEVCC {

    var HAL_61851_Listener v_HAL_61851_Listener;
    var verdicttype preConVerdict;
    var float v_timeMin := 2.0;
    var integer v_numberOfLoops := 20;
    vc_unmapSlac := false;

    // -------------- Pre Conditions-------------------------------------------------------
    f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
    for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
        preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

        //-------------- Test behavior-------------------------------------------------------
        if(preConVerdict==pass){
            f_EVCC_AC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
            if (getverdict == pass){
                f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass,
                                                 pass, inconc);
            }
        }else {
            log("PreCondition was unsuccessful.");
        }
        if(getverdict != pass) {
            break;
         } else {
             if(not(i == (v_numberOfLoops-1))) {
                 f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                 pt_HAL_61851_Internal_Port.clear;
                 pt_V2G_TCP_TLS_Port_Control.clear;
                 v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(false));
                 vc_V2G_Port_IpAddress := par_SECC_IPaddress;
                 vc_V2G_Port_PortNumber := 0;
                 vc_AC_EVSEStatus := { rCD := false,
                                       notificationMaxDelay := par_EVCC_NotificationMaxDelay,
                                       eVSENotification := none_ };
             }
         }
     }
     //------------- Post Conditions-------------------------------------------------------
     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
     f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
}

 testcase TC_EVCC_AC_VTB_Reliability_002() runs on EVCC_Tester system SystemEVCC {

     var HAL_61851_Listener v_HAL_61851_Listener;
     var verdicttype preConVerdict;
     var float v_timeMin := 15.0;
     var integer v_numberOfLoops := 2;
     vc_unmapSlac := false;

     // -------------- Pre Conditions-------------------------------------------------------
     f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
     for (var integer i:=0; i<v_numberOfLoops; i:=i + 1) {
         preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

         //-------------- Test behavior-------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_AC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
             if (getverdict == pass){
                 f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass,
                                                  pass, inconc);
             }
         }else {
             log("PreCondition was unsuccessful.");
         }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if(getverdict != pass) {
                break;
             } else {
                 if(not(i == (v_numberOfLoops-1))) {
                     f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
                     pt_HAL_61851_Internal_Port.clear;
                     pt_V2G_TCP_TLS_Port_Control.clear;
                     v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(false));
                     vc_V2G_Port_IpAddress := par_SECC_IPaddress;
                     vc_V2G_Port_PortNumber := 0;
                     vc_AC_EVSEStatus := { rCD := false,
                                           notificationMaxDelay := par_EVCC_NotificationMaxDelay,
                                           eVSENotification := none_ };
                 }
             }
         }
         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }

     testcase TC_EVCC_AC_VTB_Reliability_003() runs on EVCC_Tester system SystemEVCC {

         var HAL_61851_Listener v_HAL_61851_Listener;
         var verdicttype preConVerdict;
         var float v_timeMin := 45.0;

         // -------------- Pre Conditions-------------------------------------------------------
         f_EVCC_CMN_PR_InitConfiguration_001(v_HAL_61851_Listener,system);
         preConVerdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);

         //-------------- Test behavior--------------------------------------------------------
         if(preConVerdict==pass){
             f_EVCC_AC_TB_VTB_Reliability_001(v_HAL_61851_Listener, v_timeMin);
             if (getverdict == pass){
                 f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass,
                                                  pass, inconc);
             }
         }else {
             log("PreCondition was unsuccessful.");
         }

         //------------- Post Conditions-------------------------------------------------------
         f_EVCC_CMN_PO_InitialState_001(v_HAL_61851_Listener);
         f_EVCC_CMN_PO_ShutdownConfiguration_001(v_HAL_61851_Listener,system);
     }


}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex F
(normative)

Function specifications for supporting test execution

F.1 General information

This annex covers all pre-condition, post-condition, common behavior, and library *functions* in the ATS.

F.2 Configuration functions

```
module Configurations_15118_2 {

    import from Pixit_15118 all;
    import from Services_HAL_61851_1 all;
    import from DataStructure_HAL_61851 all;
    import from DataStructure_V2G_TCP_TLS_Port_Control all;
    import from urn_iso_15118_2_2010_AppProtocol language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from XSDAUX all;
    import from ComponentsAndPorts all;
    import from LibFunctions_15118_2 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Pics_15118 all;

    //     :::::::::::::::::::::::: Config Functions :::::::::::::::::::::::::
    function f_SECC_CMN_PR_InitConfiguration_001(out HAL_61851_Listener v_HAL_61851_Listener,
                                                 SystemSECC systemSECC) runs on SECC_Tester {

        if(PICS_CMN_CMN_CombinedTesting) {
            map(mtc:pt_SLAC_Port, systemSECC:pt_SLAC_Port);
        }
        map(mtc:pt_V2G_UDP_SDP_Port, systemSECC:pt_V2G_UDP_SDP_Port);

            v_HAL_61851_Listener := HAL_61851_Listener.create("SECC IEC 61851 Listener") alive;
            map(mtc:pt_HAL_61851_Port, systemSECC:pt_HAL_61851_Port);
            map(v_HAL_61851_Listener:pt_HAL_61851_Listener_Port,
                systemSECC:pt_HAL_61851_Listener_Port);
            connect(mtc:pt_HAL_61851_Internal_Port,
                    v_HAL_61851_Listener:pt_HAL_61851_Internal_Port);

            v_HAL_61851_Listener.start(f_SECC_HAL_61851_Listener(false, false, e_StateA));

            vc_Default_IEC_61851_ListenerBehavior :=
            activate(a_IEC_61851_ListenerBehavior(pt_HAL_61851_Internal_Port));

    }

    //     :::::::::::::::::::::::: Config Functions :::::::::::::::::::::::::
    function f_SECC_CMN_PR_InitConfiguration_002(SystemSECC systemSECC,
                                                 verdicttype v_verdict1,
                                                 verdicttype v_verdict2)
                                                 runs on SECC_Tester {

        map(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, systemSECC:pt_V2G_TCP_TLS_ALM_SECC_Port)
        param (vc_v2g_tcp_tls_parameter);

        map(mtc:pt_V2G_TCP_TLS_Port_Control, systemSECC:pt_V2G_TCP_TLS_Port_Control);

        var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_responseMessage;
        var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;

        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_connectPort));
        alt{
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] pt_V2G_TCP_TLS_Port_Control.receive(md_CMN_CMN_tcpTlsStatusRes_001(
                                                e_connectPort,?,?,*))
                                                -> value v_responseMessage {

                var boolean isClosed := v_responseMessage.result.portStatus.closed;
                ackType := v_responseMessage.ack;
            }
        }
        if(ackType == ACK_SUCCESS) {
            setverdict(v_verdict1, "TCP or TLS handshake was successful.");
        } else if(ackType == ACK_FAILED) {
            setverdict(v_verdict2, "TCP or TLS handshake was not successful.");
        }
    }

    function f_SECC_CMN_PO_ShutdownConfiguration_001(inout HAL_61851_Listener v_HAL_61851_Listener,
                                                     SystemSECC v_SystemSECC) runs on SECC_Tester {


            unmap(mtc:pt_HAL_61851_Port, v_SystemSECC:pt_HAL_61851_Port);
            unmap(v_HAL_61851_Listener:pt_HAL_61851_Listener_Port,
                  v_SystemSECC:pt_HAL_61851_Listener_Port);
            disconnect(mtc:pt_HAL_61851_Internal_Port,
                       v_HAL_61851_Listener:pt_HAL_61851_Internal_Port);
            pt_HAL_61851_Port.clear;
            v_HAL_61851_Listener.kill;

        unmap(mtc:pt_V2G_TCP_TLS_ALM_SECC_Port, v_SystemSECC:pt_V2G_TCP_TLS_ALM_SECC_Port);
        unmap(mtc:pt_V2G_TCP_TLS_Port_Control, v_SystemSECC:pt_V2G_TCP_TLS_Port_Control);
        unmap(mtc:pt_V2G_UDP_SDP_Port, v_SystemSECC:pt_V2G_UDP_SDP_Port);
        if(PICS_CMN_CMN_CombinedTesting) {
            unmap(mtc:pt_SLAC_Port, v_SystemSECC:pt_SLAC_Port);
        }
    }

    function f_SECC_PO_ShutdownConfiguration_002(SystemSECC v_SystemSECC) runs on SECC_Tester  {

        unmap(mtc:pt_SLAC_Port, v_SystemSECC:pt_SLAC_Port);
    }

    //      :::::::::::::::::::::::: Config Functions :::::::::::::::::::::::::
    function f_EVCC_CMN_PR_InitConfiguration_001(out HAL_61851_Listener v_HAL_61851_Listener,
                                                 SystemEVCC systemEVCC) runs on EVCC_Tester {

         if(PICS_CMN_CMN_CombinedTesting) {
             map(mtc:pt_SLAC_Port, systemEVCC:pt_SLAC_Port);
         }
         map(mtc:pt_V2G_UDP_SDP_Port, systemEVCC:pt_V2G_UDP_SDP_Port);
         map(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, systemEVCC:pt_V2G_TCP_TLS_ALM_EVCC_Port);
         map(mtc:pt_V2G_TCP_TLS_Port_Control, systemEVCC:pt_V2G_TCP_TLS_Port_Control);

             v_HAL_61851_Listener := HAL_61851_Listener.create("EVCC IEC 61851 Listener") alive;

             map(mtc:pt_HAL_61851_Port, systemEVCC:pt_HAL_61851_Port);
             map(v_HAL_61851_Listener:pt_HAL_61851_Listener_Port,
                  systemEVCC:pt_HAL_61851_Listener_Port);
             connect(mtc:pt_HAL_61851_Internal_Port,
                     v_HAL_61851_Listener:pt_HAL_61851_Internal_Port);

             v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(false));

             vc_Default_IEC_61851_ListenerBehavior :=
             activate(a_IEC_61851_ListenerBehavior(pt_HAL_61851_Internal_Port));
     }

    //     :::::::::::::::::::::::: Config Functions :::::::::::::::::::::::::
    function f_EVCC_CMN_PR_InitConfiguration_002(SystemEVCC systemEVCC,
                                                 in verdicttype v_vct)
                                                 runs on EVCC_Tester  {

        var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_responseMessage;
        var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;

        map(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, systemEVCC:pt_V2G_TCP_TLS_ALM_EVCC_Port)
            param (vc_v2g_tcp_tls_parameter);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_connectPort));
        alt{
             [] pt_V2G_TCP_TLS_Port_Control.receive(md_CMN_CMN_tcpTlsStatusRes_001(
                                          e_connectPort,?,?,*))
                                          -> value v_responseMessage {

                 var boolean isClosed := v_responseMessage.result.portStatus.closed;
                 ackType := v_responseMessage.ack;
                 if(ackType == ACK_SUCCESS) {
                    setverdict(pass, "TCP or TLS server could be started.");
                 } else if(ackType == ACK_FAILED) {
                    setverdict(v_vct, "TCP or TLS server could not be started.");
                 }
             }
             [] pt_V2G_TCP_TLS_Port_Control.receive {
                setverdict(v_vct, "TCP or TLS server could not be started.");
             }
         }

         pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                     e_getPort));
         alt{
            [] pt_V2G_TCP_TLS_Port_Control.receive(md_CMN_CMN_tcpTlsStatusRes_002(
                                         e_getPort,ACK_SUCCESS,
                                         (49152 .. 65535)))
                                         -> value v_responseMessage {

                vc_V2G_Port_PortNumber := v_responseMessage.result.portNumber.intValue;
            }
         }
    }

    function f_EVCC_CMN_PO_ShutdownConfiguration_001(inout HAL_61851_Listener v_HAL_61851_Listener,
                                                     SystemEVCC v_SystemEVCC) runs on EVCC_Tester {

            unmap(mtc:pt_HAL_61851_Port, v_SystemEVCC:pt_HAL_61851_Port);
            unmap(v_HAL_61851_Listener:pt_HAL_61851_Listener_Port,
                  v_SystemEVCC:pt_HAL_61851_Listener_Port);
            disconnect(mtc:pt_HAL_61851_Internal_Port,
                       v_HAL_61851_Listener:pt_HAL_61851_Internal_Port);
            pt_HAL_61851_Port.clear;
            v_HAL_61851_Listener.kill;


        unmap(mtc:pt_V2G_TCP_TLS_ALM_EVCC_Port, v_SystemEVCC:pt_V2G_TCP_TLS_ALM_EVCC_Port);
        unmap(mtc:pt_V2G_TCP_TLS_Port_Control, v_SystemEVCC:pt_V2G_TCP_TLS_Port_Control);
        unmap(mtc:pt_V2G_UDP_SDP_Port, v_SystemEVCC:pt_V2G_UDP_SDP_Port);
        if(PICS_CMN_CMN_CombinedTesting) {
            unmap(mtc:pt_SLAC_Port, v_SystemEVCC:pt_SLAC_Port);
        }
    }

    function f_EVCC_PO_ShutdownConfiguration_002(SystemEVCC v_SystemEVCC) runs on EVCC_Tester  {

        unmap(mtc:pt_SLAC_Port, v_SystemEVCC:pt_SLAC_Port);
    }
}
```

F.3 Pre-condition functions

F.3.1 SECC functions

```
module PreConditions_SECC_15118_2 {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118_2 all;
    import from Pixit_15118_2 all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Pics_15118 all;
import from Timer_15118_2 all;
import from Timer_15118_3 all;
import from DataStructure_SDP all;
import from ComponentsAndPorts all;
import from DataStructure_HAL_61851 all;
import from TestBehavior_SECC_SDP all;
import from TestBehavior_SECC_CommonBehavior all;
import from LibFunctions_15118_2 all;
import from TestBehavior_SECC_SupportedAppProtocol all;
import from TestBehavior_SECC_SessionSetup all;
import from TestBehavior_SECC_ServiceDiscovery all;
import from TestBehavior_SECC_Authorization all;
import from TestBehavior_SECC_PowerDelivery all;
import from TestBehavior_SECC_PaymentServiceSelection all;
import from TestBehavior_SECC_ServiceDetail all;
import from TestBehavior_SECC_PaymentDetails all;
import from TestBehavior_SECC_CertificateInstallation all;
import from TestBehavior_SECC_CertificateUpdate all;
import from TestBehavior_SECC_ChargeParameterDiscovery all;
import from TestBehavior_SECC_ChargingStatus all;
import from TestBehavior_SECC_MeteringReceipt all;
import from TestBehavior_SECC_CableCheck all;
import from TestBehavior_SECC_PreCharge all;
import from TestBehavior_SECC_CurrentDemand all;
import from TestBehavior_SECC_WeldingDetection all;
import from TestBehavior_SECC_SessionStop all;
import from Configurations_15118_2 all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from Services_Security all;
import from Services_SECC_PLCLinkStatus all;

import from Pixit_15118 all;
import from Timer_15118 all;
import from PreConditions_SECC_15118_3 {
    function f_SECC_CMN_PR_CmAmpMap_001;
}
import from TTlibrary_Logging all;
import from urn_iso_15118_2_2013_X509Def all;
import from Services_MessageBox all;

import from TestBehavior_SECC_CmSetKey all;

//  SECC
function f_SECC_CMN_PR_GetSecurityData_001(out HAL_61851_Listener
                                      v_HAL_61851_Listener)
                                      runs on SECC_Tester
                                      return verdicttype {

    var verdicttype verdict :=pass;

    if(PICS_CMN_CMN_IdentificationMode == pnC or PIXIT_SECC_CMN_TLS) {

        if(PIXIT_CMN_CMN_PkiName == PKI_Ext) {

            vc_v2gRootCert :=
            fx_loadCertificate(PIXIT_CMN_CMN_PkiName, cc_crt_v2g_root_valid);

            vc_IdCertificateMap := {
                [0] := {cc_crt_oem_leaf_valid,
                        fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                           cc_crt_oem_leaf_valid)}
            };

            vc_IdPrivateKeyMap := {
                [0] := {cc_crt_oem_leaf_valid,
                        fx_loadPrivateKey(cc_crt_oem_leaf_valid)}
            };

        } else if(PIXIT_CMN_CMN_PkiName == PKI_6) {

            if(not PIXIT_SECC_CMN_TLS) {
                setverdict(inconc, "Invalid PKI was selected. PKI-6 can only be used for " &
                                   "use case EIM with TLS support. Please check PICS and " &
                                   "PIXIT configuration.");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        } else {

            vc_peRootCert :=
            fx_loadCertificate(PIXIT_CMN_CMN_PkiName, cc_crt_pe_root_valid);
        }

    } else {

        vc_v2gRootCert :=
        fx_loadCertificate(PIXIT_CMN_CMN_PkiName, cc_crt_v2g_root_valid);

        vc_IdCertificateMap := {
            [0] := {cc_crt_oem_leaf_valid,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_oem_leaf_valid)},
            [1] := {cc_crt_oem_leaf_expired,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_oem_leaf_expired)},
            [2] := {cc_crt_oem_leaf_invalid_content,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_oem_leaf_invalid_content)},
            [3] := {cc_crt_contract_leaf_expires,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_contract_leaf_expires)},
               [4] := {cc_crt_mo_sub1_valid,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_mo_sub1_valid)},
               [5] := {cc_crt_mo_sub2_valid,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_mo_sub2_valid)},
               [6] := {cc_crt_contract_leaf_expired,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_contract_leaf_expired)},
            [7] := {cc_crt_contract_leaf_invalid_content,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_contract_leaf_invalid_content)},
            [8] := {cc_crt_contract_leaf_expired_issuer_sub1,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_contract_leaf_expired_issuer_sub1)},
               [9] := {cc_crt_contract_leaf_expired_issuer_sub2,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_contract_leaf_expired_issuer_sub2)},
            [10] := {cc_crt_mo_sub1_expired,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_mo_sub1_expired)},
            [11] := {cc_crt_mo_sub2_expired,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_mo_sub2_expired)},
            [12] := {cc_crt_contract_leaf_wrong_emaid,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_contract_leaf_wrong_emaid)},
            [13] := {cc_crt_contract_leaf_invalid_emaid,
                    fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                       cc_crt_contract_leaf_invalid_emaid)}

        };

        vc_IdPrivateKeyMap := {
            [0] := {cc_crt_oem_leaf_valid,
                    fx_loadPrivateKey(cc_crt_oem_leaf_valid)},
            [1] := {cc_crt_oem_leaf_expired,
                    fx_loadPrivateKey(cc_crt_oem_leaf_expired)},
            [2] := {cc_crt_oem_leaf_invalid_content,
                    fx_loadPrivateKey(cc_crt_oem_leaf_invalid_content)},
            [3] := {cc_crt_contract_leaf_expires,
                    fx_loadPrivateKey(cc_crt_contract_leaf_expires)},
            [4] := {cc_crt_contract_leaf_wrong_emaid,
                    fx_loadPrivateKey(cc_crt_contract_leaf_wrong_emaid)},
            [5] := {cc_crt_contract_leaf_invalid_emaid,
                    fx_loadPrivateKey(cc_crt_contract_leaf_invalid_emaid)},
            [6] := {cc_crt_contract_leaf_expired,
                    fx_loadPrivateKey(cc_crt_contract_leaf_expired)},
            [7] := {cc_crt_contract_leaf_invalid_content,
                    fx_loadPrivateKey(cc_crt_contract_leaf_invalid_content)}
        };
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
    }

    return verdict;
}

function f_SECC_CMN_PR_PLCLinkStatusCombinedTesting_001(HAL_61851_Listener
                                                        v_HAL_61851_Listener)
                                                        runs on SECC_Tester
                                                        return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_GetSecurityData_001();

    if(PICS_CMN_CMN_CombinedTesting and verdict == pass) {
        verdict := f_SECC_CMN_PR_CmAmpMap_001(v_HAL_61851_Listener);
    }
    vc_DepartureTime := 10800; // 3h
    return getverdict;
}


function f_SECC_CMN_PR_StateA_001(HAL_61851_Listener v_HAL_61851_Listener)
                                  runs on SECC_Tester return verdicttype {

    var float v_UnplugTimeRnd := 0.015;
    var verdicttype verdict := f_SECC_CMN_PR_PLCLinkStatusCombinedTesting_001(
                                                          v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting and verdict == pass){

        f_SECC_changeValidStateCondition(e_StateA);
        f_SECC_setState(e_StateA,v_HAL_61851_Listener);
        pt_HAL_61851_Port.clear;
        sleep(v_UnplugTimeRnd);
        f_SECC_setProximityOpenCircuit(true);
    }
    return getverdict;
}

function f_SECC_CMN_PR_SetProximityPilot_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                             runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_StateA_001(v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting and verdict == pass){

        if(PICS_CMN_CMN_PlugType == type1) {
            f_SECC_setProximityResistance(cc_ev_proximity_type1);
        }
        f_SECC_setProximityVoltage(true);

        v_HAL_61851_Listener.stop;
        v_HAL_61851_Listener.start(f_SECC_HAL_61851_Listener(true, true, e_StateA));

        vc_validDutyCycleLowerBound1  := par_validDutyCycleLowerBound;
        vc_validDutyCycleUpperBound1  := par_validDutyCycleUpperBound;
        vc_validDutyCycleLowerBound2 := par_validDutyCycleLowerBound;
        vc_validDutyCycleUpperBound2 := par_validDutyCycleUpperBound;

        if(PICS_CMN_CMN_ChargingMode == aC){
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            vc_validDutyCycleLowerBound2 := vc_DutyCycleOscOff;
            vc_validDutyCycleUpperBound2 := vc_DutyCycleOscOff;
        }
    }
    return getverdict;
}

function f_SECC_CMN_PR_StateB_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                  runs on SECC_Tester return verdicttype {

    timer t_plugin;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        var verdicttype verdict := f_SECC_CMN_PR_SetProximityPilot_001(v_HAL_61851_Listener);

        if(not PICS_CMN_CMN_CombinedTesting and verdict == pass){
            f_SECC_changeValidStateCondition(e_StateValid_Matching);
            t_plugin.start(par_CMN_waitForNextHAL);
            f_SECC_setState(e_StateB,v_HAL_61851_Listener);
            alt {
                [] a_SECC_PluginDetection(pt_HAL_61851_Internal_Port){
                    t_plugin.stop;
                }
                [] t_plugin.timeout {
                    var IEC_61851_States v_state := f_SECC_getMeasState();
                    var float v_dutyCycle := f_SECC_getMeasDutyCycle();
                    var float v_frequency := f_SECC_getMeasFrequency();
                    log("Measured CP State: " & toString<IEC_61851_States>(v_state));
                    log("Measured duty cycle: " & toString<float>(v_dutyCycle) & "%");
                    log("Measured frequency: " & toString<float>(v_frequency) & "Hz");
                    if(v_state == e_StateE or v_state == e_StateF) {
                       setverdict(inconc,        "No        plug-in        detected,        '"        &
toString<IEC_61851_States>(v_state) &
                                            " 'was measured after 't_plugin' timeout. Please check if "
&
                                            "the charging cable is connected correctly or " &
                                            "if the EVSE signals a valid state.");
                    } else {
                       setverdict(inconc, "Internal error. Unexpected CP State " &
                                            "was measured after plugin.");
                    }
                }
            }
        }
        // start CommunicationSetup_Timer
        tc_V2G_EVCC_CommunicationSetup_Timer.start;
        tc_V2G_SECC_CommunicationSetup_Timer.start;
        verdict := getverdict;
        return verdict;

    }

    function f_SECC_CMN_PR_Binding_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                       runs on SECC_Tester return verdicttype {

        var verdicttype verdict := f_SECC_CMN_PR_StateB_001(v_HAL_61851_Listener);

        if(vc_unmapSlac) {
            f_SECC_PO_ShutdownConfiguration_002(system);
        }

        //  SECC_Binding Behavior
        if ( verdict == pass ) {
            var Security_TYPE v_security := cc_hexTls;
            if(PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_SECC_CMN_TLS == false) {
                v_security := cc_hexTcp;
            }
            verdict := f_SECC_CMN_TB_VTB_SDP_001(v_security, inconc);
            if(vc_Security == cc_hexTcp) {
                vc_v2g_tcp_tls_parameter := md_CMN_CMN_TcpParameter_001(vc_V2G_Port_IpAddress,
                                                                        vc_V2G_Port_PortNumber,
                                                                        omit);
            } else if(vc_Security == cc_hexTls) {
                vc_v2g_tcp_tls_parameter := md_CMN_CMN_TlsParameter_002(
                                            vc_V2G_Port_IpAddress,
                                            vc_V2G_Port_PortNumber,
                                            omit, vc_cipherSuites,
                                            omit, vc_v2gRootCert);
            } else {log("Invalid security type was chosen.")};
        }
        return verdict;
    }

    function f_SECC_CMN_PR_SupportedAppProtocol_001(out HAL_61851_Listener
                                                    v_HAL_61851_Listener)
                                                    runs on SECC_Tester return verdicttype {

        var verdicttype verdict;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        verdict := f_SECC_CMN_PR_Binding_001(v_HAL_61851_Listener);
    // SECC_SupportedAppProtocol Behavior
    if (verdict == pass) {
        f_SECC_CMN_PR_InitConfiguration_002(system, pass, inconc);
        verdict := getverdict;
            if (verdict == pass) {
            verdict := f_SECC_CMN_TB_VTB_SupportedAppProtocol_001(inconc);
            //clear SDP Port
            pt_V2G_UDP_SDP_Port.clear;
            vc_Default_SDP_Message := activate(a_SDP_Message());
        }
    }
    return verdict;
}

function f_SECC_CMN_PR_SessionSetup_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                        runs on SECC_Tester return verdicttype {

   var verdicttype verdict :=
   f_SECC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);
   // SECC_SessionSetup Behavior
   if (verdict == pass) {
       verdict := f_SECC_CMN_TB_VTB_SessionSetup_001(inconc);
       tc_V2G_EVCC_CommunicationSetup_Timer.stop;
       tc_V2G_SECC_CommunicationSetup_Timer.stop;
   }
   return verdict;
 }

function f_SECC_CMN_PR_ServiceDiscovery_001(out HAL_61851_Listener
                                            v_HAL_61851_Listener)
                                            runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);
   // SECC_ServiceDiscovery Behavior
   if (verdict == pass) {
       verdict := f_SECC_CMN_TB_VTB_ServiceDiscovery_001(inconc);
   }
   return verdict;
}

function f_SECC_CMN_PR_ServiceDetail_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                         runs on SECC_Tester return verdicttype {

   var verdicttype verdict := f_SECC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);
   // SECC_ServiceDetail Behavior
   if (verdict == pass and (ispresent(vc_serviceList.service_list) and
                               PIXIT_SECC_CMN_VAS == serviceDetail)) {
           if(not(PICS_CMN_CMN_IdentificationMode == eIM and
              PIXIT_SECC_CMN_TLS == false)) {
              verdict := f_SECC_CMN_TB_VTB_ServiceDetail_001(inconc);
           }
   }
   return verdict;
 }

function f_SECC_CMN_PR_PaymentServiceSelection_001(out HAL_61851_Listener
                                                   v_HAL_61851_Listener)
                                                   runs on SECC_Tester
                                                   return verdicttype {

   var verdicttype verdict := f_SECC_CMN_PR_ServiceDetail_001(v_HAL_61851_Listener);
   // SECC_PaymentServiceSelection Behavior
   if (verdict == pass) {
       verdict := f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(inconc);
   }
   return verdict;
 }

function f_SECC_CMN_PR_CertificateInstallation_001(out HAL_61851_Listener
                                                   v_HAL_61851_Listener)
                                                   runs on SECC_Tester
                                                   return verdicttype {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   var verdicttype verdict;

          verdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   // SECC_CertificateInstallation Behavior
   if (verdict == pass) {
       verdict := f_SECC_CMN_TB_VTB_CertificateInstallation_001(inconc);
   }
   return verdict;
}

function f_SECC_CMN_PR_CertificateUpdate_001(out HAL_61851_Listener
                                             v_HAL_61851_Listener)
                                             runs on SECC_Tester
                                             return verdicttype {

    var verdicttype verdict;

        verdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   // SECC_CertificateUpdate Behavior
   if (verdict == pass) {
       verdict := f_SECC_CMN_TB_VTB_CertificateUpdate_001(inconc);
   }
   return verdict;
}

function f_SECC_CMN_PR_PaymentDetails_001(out HAL_61851_Listener
                                          v_HAL_61851_Listener)
                                          runs on SECC_Tester
                                          return verdicttype {

   var verdicttype verdict;

   if(PIXIT_SECC_CMN_CertificateHandling == certUpdate and
      PICS_CMN_CMN_IdentificationMode == pnC) {
      verdict := f_SECC_CMN_PR_CertificateUpdate_001(v_HAL_61851_Listener);
   }
   if(PIXIT_SECC_CMN_CertificateHandling == certInstall and
      PICS_CMN_CMN_IdentificationMode == pnC){
      verdict := f_SECC_CMN_PR_CertificateInstallation_001(v_HAL_61851_Listener);
   }
   if((PIXIT_SECC_CMN_CertificateHandling == none_) and
          PICS_CMN_CMN_IdentificationMode == pnC){

            verdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   }
   // SECC_PaymentDetails Behavior
   if (verdict == pass) {
       verdict := f_SECC_CMN_TB_VTB_PaymentDetails_001(inconc);
   }
   return verdict;
}

function f_SECC_CMN_PR_Authorization_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                         runs on SECC_Tester return verdicttype {

   var verdicttype verdict;

   if(PICS_CMN_CMN_IdentificationMode == pnC) {
       verdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);
   }
   if(PICS_CMN_CMN_IdentificationMode == eIM){
            verdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);

   }
   // SECC_Authorization Behavior
   if (verdict == pass) {
       if(PICS_CMN_CMN_IdentificationMode == eIM){
           verdict := f_SECC_CMN_TB_VTB_Authorization_001(inconc);
       }
       if(PICS_CMN_CMN_IdentificationMode == pnC){
           verdict := f_SECC_CMN_TB_VTB_Authorization_002(inconc);
       }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   }
   return verdict;
}

function f_SECC_CMN_PR_Authorization_002(out HAL_61851_Listener v_HAL_61851_Listener)
                                         runs on SECC_Tester return verdicttype {

   var verdicttype verdict;

   if(PICS_CMN_CMN_IdentificationMode == pnC) {
       verdict := f_SECC_CMN_PR_PaymentDetails_001(v_HAL_61851_Listener);
   }
   if(PICS_CMN_CMN_IdentificationMode == eIM){
       verdict := f_SECC_CMN_PR_PaymentServiceSelection_001(v_HAL_61851_Listener);
   }
   // SECC_Authorization Behavior
   if (verdict == pass) {
       if(PICS_CMN_CMN_IdentificationMode == eIM){
           verdict := f_SECC_CMN_TB_VTB_Authorization_001(inconc);
       }
       if(PICS_CMN_CMN_IdentificationMode == pnC){
           verdict := f_SECC_CMN_TB_VTB_Authorization_002(inconc);
       }
   }
   return verdict;
}

//  PreConditions for AC-TestCases

function f_SECC_AC_PR_ChargeParameterDiscovery_001(out HAL_61851_Listener
                                                    v_HAL_61851_Listener)
                                                    runs on SECC_Tester return verdicttype {

   var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

   // SECC_ChargeParameterDiscovery Behavior
   if (verdict == pass) {
           if(PIXIT_SECC_CMN_SalesTariff == unknown) {
            verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
           } else {verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
   }
   if(getverdict == pass and PIXIT_CMN_AC_PowerFlow) {
        f_getACMaxCurrent();
   }
   return verdict;
}

function f_SECC_AC_PR_PowerDeliveryStart_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                             runs on SECC_Tester return verdicttype {

   var verdicttype verdict := f_SECC_AC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

   // SECC_PowerDeliveryStart Behavior
   if (verdict == pass) {
       verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
                                                     inconc);
   }
   return verdict;
}

function f_SECC_AC_PR_ChargingStatus_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                         runs on SECC_Tester return verdicttype {

    var verdicttype verdict := f_SECC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);

    // SECC_ChargingStatus Behavior
    if (verdict == pass) {
         verdict := f_SECC_AC_TB_VTB_ChargingStatus_001(inconc);
    }
    return verdict;
}

function f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(out HAL_61851_Listener
                                                               v_HAL_61851_Listener)
                                                               runs on SECC_Tester
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                                    return verdicttype {

            var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);
            if (verdict == pass) {
                var integer v_loopCounter := 0;
                var integer renegotiationLoopInd := PICS_CMN_CMN_RenegotiationLoopIndication;
                if(not PICS_CMN_CMN_Renegotiation) {renegotiationLoopInd := -1};
                while (vc_ChargeProgress != stop_){
                    // ChargeParameterDiscovery
                    if (verdict == pass) {
                            if (vc_ChargeProgress == start_ and v_loopCounter == 0) {
                              if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                                  verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
                              } else {verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc);
}
                              if(getverdict == pass and PIXIT_CMN_AC_PowerFlow) {
                                  f_getACMaxCurrent();
                              }
                            }
                    } else { return verdict; }

                    // PowerDelivery
                    if (verdict == pass) {
                          if (vc_ChargeProgress == start_) {
                                 verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(start_,
                                                                                v_HAL_61851_Listener,
                                                                                inconc);

                      }
                    } else { return verdict;}

                    while(vc_EVSENotification == none_ and v_loopCounter <
                            PICS_CMN_CMN_LoopCounter and v_loopCounter != renegotiationLoopInd) {
                        // ChargingStatus
                        if (verdict == pass) {
                            if(true){ verdict := f_SECC_AC_TB_VTB_ChargingStatus_001(inconc); }}
                            else { return verdict;}

                            v_loopCounter := v_loopCounter + 1;
                          log("-------------------------------------------------------------");
                          log("Number of received ChargingStatusRes messages: " &
                              toString<integer>(v_loopCounter));
                          log("-------------------------------------------------------------");


                        // MeteringReceipt
                        if (verdict == pass) {
                            if(vc_receiptRequired and PICS_CMN_CMN_IdentificationMode == pnC){
                                verdict := f_SECC_AC_TB_VTB_MeteringReceipt_001(inconc); }}
                            else { return verdict; }

                        if (verdict == pass) {
                            sleep(par_SECC_chargingLoop_pause);
                        }
                     }

                    if (vc_EVSENotification == reNegotiation or
                        v_loopCounter == renegotiationLoopInd and PICS_CMN_CMN_Renegotiation){
                        if(verdict == pass) {vc_ChargeProgress := renegotiate;}
                        else { return verdict; }
                            if(verdict == pass) {
                                 verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(renegotiate,
                                                                                v_HAL_61851_Listener,
                                                                                inconc)
                            } else { return verdict;}
                            if(verdict == pass) {
                                vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 10 };
                                if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                                    verdict :=
                                    f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
                                } else {verdict :=
                                          f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
                                if(getverdict == pass and PIXIT_CMN_AC_PowerFlow) {
                                   f_getACMaxCurrent();
                                }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        if(verdict == pass) {
                            vc_ChargeProgress := start_; renegotiationLoopInd := 0;
                            vc_EVSENotification := none_}
                    } else { return verdict; }
            }
            if(v_loopCounter == PICS_CMN_CMN_LoopCounter or
                 vc_EVSENotification == stopCharging){vc_ChargeProgress := stop_;}
        }
    }
    return verdict;
}

function f_SECC_AC_PR_ChargingStatusOrMeteringReceiptRenegotiation_001(out HAL_61851_Listener
                                                                         v_HAL_61851_Listener)
                                                                         runs on SECC_Tester
                                                                         return verdicttype {

    var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);
    if (verdict == pass) {

        while (vc_ChargeProgress != renegotiate){
            // ChargeParameterDiscovery
            if (verdict == pass and vc_ChargeProgress == start_) {
              if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                  verdict :=
                  f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
              } else {verdict :=
                      f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
            } else { return verdict; }
            if(getverdict == pass and PIXIT_CMN_AC_PowerFlow) {
               f_getACMaxCurrent();
            }
            // PowerDelivery
            if (verdict == pass and vc_ChargeProgress == start_) {
                 verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
                                                                inconc);
            } else { return verdict; }

            var integer v_loopCounter := 0;
            while(vc_EVSENotification == none_ and
                  v_loopCounter < PICS_CMN_CMN_LoopCounter and
                  v_loopCounter != PICS_CMN_CMN_RenegotiationLoopIndication) {
                // ChargingStatus
                if (verdict == pass) {
                      verdict := f_SECC_AC_TB_VTB_ChargingStatus_001(inconc);
                } else { return verdict; }

                    v_loopCounter := v_loopCounter + 1;
                  log("-------------------------------------------------------------");
                  log("Number of received ChargingStatusRes messages: " &
                      toString<integer>(v_loopCounter));
                  log("-------------------------------------------------------------");


                // MeteringReceipt
                if (verdict == pass) {
                    if(vc_receiptRequired and
                    PICS_CMN_CMN_IdentificationMode == pnC){
                        verdict := f_SECC_AC_TB_VTB_MeteringReceipt_001(inconc); }}
                else { return verdict; }
                if (verdict == pass) {
                    sleep(par_SECC_chargingLoop_pause);
                }
             }

            if (vc_EVSENotification == reNegotiation or
                v_loopCounter == PICS_CMN_CMN_RenegotiationLoopIndication and
                  PICS_CMN_CMN_Renegotiation){
                if(verdict == pass) {
                    vc_ChargeProgress := renegotiate;}
                else { return verdict; }
            }
        }
    }
    return verdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  }

function f_SECC_AC_PR_PowerDeliveryRenegotiation_001(out HAL_61851_Listener
                                                     v_HAL_61851_Listener)
                                                     runs on SECC_Tester return verdicttype {

       var verdicttype verdict;
       verdict :=
       f_SECC_AC_PR_ChargingStatusOrMeteringReceiptRenegotiation_001(v_HAL_61851_Listener);

       // f_SECC_PowerDeliveryStopAC Behavior
       if (verdict == pass) {
             verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(renegotiate, v_HAL_61851_Listener,
                                                           inconc);
       }
       return verdict;
}

function f_SECC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(out HAL_61851_Listener
                                                                v_HAL_61851_Listener)
                                                                runs on SECC_Tester
                                                                return verdicttype {

       var verdicttype verdict;
       verdict := f_SECC_AC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

       // f_SECC_ChargeParameterDiscovery Behavior
       if (verdict == pass) {
        vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 10 };
        if(PIXIT_SECC_CMN_SalesTariff == unknown) {
            verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
        } else {verdict := f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
       }
       if(getverdict == pass and PIXIT_CMN_AC_PowerFlow) {
          f_getACMaxCurrent();
       }
       return verdict;
}

function f_SECC_AC_PR_PowerDeliveryStop_001(out HAL_61851_Listener
                                            v_HAL_61851_Listener)
                                            runs on SECC_Tester return verdicttype {

     var verdicttype verdict;

     verdict := f_SECC_AC_PR_ChargingStatusOrMeteringReceiptStop_001(v_HAL_61851_Listener);

     // SECC_f_SECC_PowerDeliveryStopAC Behavior
     if (verdict == pass) {
           verdict := f_SECC_AC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                                         inconc);
     }
     return verdict;
}

function f_SECC_AC_PR_SessionStop_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                      runs on SECC_Tester return verdicttype {

   var verdicttype verdict;

   verdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

   // SECC_WeldingDetection Behavior
   if (verdict == pass) {
         verdict := f_SECC_CMN_TB_VTB_SessionStop_001(pause, v_HAL_61851_Listener,
                                                      inconc);
   }
   return verdict;
}

function f_SECC_AC_PR_SessionStop_002(out HAL_61851_Listener v_HAL_61851_Listener)
                                      runs on SECC_Tester return verdicttype {

   var verdicttype verdict;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        verdict := f_SECC_AC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener);

        // SECC_WeldingDetection Behavior
        if (verdict == pass) {
              verdict := f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener,
                                                           inconc);
        }
        return verdict;
    }

    //  PreConditions for DC-TestCases

    function f_SECC_DC_PR_ChargeParameterDiscovery_001(out HAL_61851_Listener
                                                        v_HAL_61851_Listener)
                                                        runs on SECC_Tester
                                                        return verdicttype {

        var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

        // SECC_ChargeParameterDiscovery Behavior
        if (verdict == pass) {
          if(PIXIT_SECC_CMN_SalesTariff == unknown) {
              verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
          } else {verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
        }
        return verdict;
    }

    function f_SECC_DC_PR_CableCheck_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                         runs on SECC_Tester return verdicttype {

        var verdicttype verdict :=
        f_SECC_DC_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);

        // SECC_CableCheck Behavior
        if (verdict == pass) {
            verdict := f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, inconc);
        }
        return verdict;
    }

    function f_SECC_DC_PR_PreCharge_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                        runs on SECC_Tester return verdicttype {

          var verdicttype verdict := f_SECC_DC_PR_CableCheck_001(v_HAL_61851_Listener);

          // SECC_PreCharge Behavior
          if (verdict == pass) {
               vc_EVTargetVoltage := par_EVBatteryVoltage;
and vc_tcRunning
               verdict := f_SECC_DC_TB_VTB_PreCharge_001(inconc);
          }

          return verdict;
    }

    function f_SECC_DC_PR_PowerDeliveryStart_001(out HAL_61851_Listener
                                                 v_HAL_61851_Listener)
                                                 runs on SECC_Tester return verdicttype {

        var verdicttype verdict := f_SECC_DC_PR_PreCharge_001(v_HAL_61851_Listener);

        // SECC_PowerDeliveryStart Behavior
        if (verdict == pass) {
           verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
                                                          false, true, inconc);
        }
        return verdict;
    }

    function f_SECC_DC_PR_CurrentDemand_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                            runs on SECC_Tester return verdicttype {

        var verdicttype verdict :=
        f_SECC_DC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    if (verdict == pass) {
       // SECC_PowerDeliveryStart Behavior
       verdict := f_SECC_DC_TB_VTB_CurrentDemand_001(inconc);
    }
    return verdict;
}

function f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(out HAL_61851_Listener
                                                              v_HAL_61851_Listener)
                                                              runs on SECC_Tester
                                                              return verdicttype {

  var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);

  if (verdict == pass) {
      var integer v_loopCounter := 0;
      var integer moduloValueSOC;

      var integer renegotiationLoopInd := PICS_CMN_CMN_RenegotiationLoopIndication;
      if(not PICS_CMN_CMN_Renegotiation) {renegotiationLoopInd := -1};
      if(par_EVRESSSOC == v_fullBatteryVal) {
         v_fullBattery := true;
      }
      while (vc_ChargeProgress != stop_){

          // ChargeParameterDiscovery
          if (verdict == pass) {
               if (vc_ChargeProgress == start_ and v_loopCounter == 0) {
                if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                    verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
                } else {verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
               }
          } else { return verdict; }

          // CableCheck
          if (verdict == pass) {
            if(vc_ChargeProgress == start_) {
              verdict := f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, inconc);
            }
          } else { return verdict; }

          // PreCharge
          if (verdict == pass) {
            if(vc_ChargeProgress == start_) {
              vc_EVTargetVoltage := par_EVBatteryVoltage;
              verdict := f_SECC_DC_TB_VTB_PreCharge_001(inconc);
            }

          } else { return verdict; }

          // PowerDelivery
          if (verdict == pass) {
            if(vc_ChargeProgress == start_ ) {
                verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
                                                              false, true, inconc);
            }
          } else { return verdict;}

          vc_EVTargetCurrent.value_ := float2int(par_InitialCurrentChargingLoop);
          vc_EVTargetCurrent.multiplier := 0;
          log("The initial target current is set to 'par_InitialCurrentChargingLoop'.");

          while(vc_EVSENotification == none_ and
                v_loopCounter < PICS_CMN_CMN_LoopCounter and
                v_loopCounter != renegotiationLoopInd) {

              if(v_loopCounter + 1 == PICS_CMN_CMN_LoopCounter) {
                 vc_EVTargetCurrent.value_ := 0;
              }
              if(v_fullBattery)    {
                 vc_EVTargetCurrent.value_ := 0;
                 log("The State of Charge is equal to 100%, an EVTargetCurrent of 0A is used.");
              }
              // CurrentDemand
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if (verdict == pass) {
              if(true){ verdict := f_SECC_DC_TB_VTB_CurrentDemand_001(inconc); }}
            else { return verdict;}

            log("------------------------------------------------------------");
            log("Number of received CurrentDemandRes messages: " &
                  toString<integer>(v_loopCounter + 1));
            log("------------------------------------------------------------");


            // MeteringReceipt
            if (verdict == pass) {
              if(vc_receiptRequired and PICS_CMN_CMN_IdentificationMode == pnC) {
                  verdict := f_SECC_DC_TB_VTB_MeteringReceipt_001(inconc); }
            } else { return verdict; }


            if (verdict == pass) {
                sleep(par_SECC_chargingLoop_pause);
                v_loopCounter := v_loopCounter + 1;
            }
        }
        if((v_loopCounter == renegotiationLoopInd-1) and PICS_CMN_CMN_Renegotiation) {
          vc_EVTargetCurrent.value_ := 0;
        }

        if (vc_EVSENotification == reNegotiation or
            v_loopCounter == renegotiationLoopInd and PICS_CMN_CMN_Renegotiation) {
            if(verdict == pass) {
                vc_ChargeProgress := renegotiate;
                } else { return verdict; }
                if (verdict == pass) {
                      verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(renegotiate,
                                                                    v_HAL_61851_Listener,
                                                                    false, true, inconc)
                } else { return verdict;}
                if (verdict == pass) {
                    vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 50 };
                    if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                        verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
                    } else {verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
                    if (verdict == pass) {
                      vc_EVTargetCurrent := { multiplier := 0, unit := a, value_ := 1   };
                      vc_ChargeProgress := start_; renegotiationLoopInd := 0;
                      vc_EVSENotification := none_}
                } else { return verdict; }
        }
        if(v_loopCounter == PICS_CMN_CMN_LoopCounter or
           vc_EVSENotification == stopCharging){vc_ChargeProgress := stop_;}
      }
    }
    return verdict;
}

function f_SECC_DC_PR_CurrentDemandOrMeteringReceiptRenegotiation_001(out HAL_61851_Listener
                                                                    v_HAL_61851_Listener)
                                                                    runs on SECC_Tester
                                                                    return verdicttype {

  var verdicttype verdict := f_SECC_CMN_PR_Authorization_001(v_HAL_61851_Listener);
  if (verdict == pass) {

      while (vc_ChargeProgress != renegotiate){

          // ChargeParameterDiscovery
          if (verdict == pass and vc_ChargeProgress == start_) {
            if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
            } else {verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
          } else { return verdict; }

          // CableCheck
          if (verdict == pass and vc_ChargeProgress == start_) {
              verdict := f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, inconc);
          } else { return verdict; }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
// PreCharge
if (verdict == pass and vc_ChargeProgress == start_) {
    vc_EVTargetVoltage := par_EVBatteryVoltage;
    verdict := f_SECC_DC_TB_VTB_PreCharge_001(inconc);
} else { return verdict; }

// PowerDelivery
if (verdict == pass and vc_ChargeProgress == start_) {
      verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(start_,
                                                    v_HAL_61851_Listener,
                                                    false, true, inconc);
} else { return verdict;}

var integer v_loopCounter := 0;
while(vc_EVSENotification == none_ and
      v_loopCounter < PICS_CMN_CMN_LoopCounter and
      v_loopCounter != PICS_CMN_CMN_RenegotiationLoopIndication){
    if(v_loopCounter + 1 == PICS_CMN_CMN_LoopCounter) {
       vc_EVTargetCurrent.value_ := 0;
    }
    if(v_fullBattery)    {
       vc_EVTargetCurrent.value_ := 0;
       log("The State of Charge is equal to 100%, an EVTargetCurrent of 0A is used.");
    }
    // CurrentDemand
    if (verdict == pass) {
      if(true){ verdict := f_SECC_DC_TB_VTB_CurrentDemand_001(inconc); }}
    else { return verdict;}

    log("-------------------------------------------------------------");
    log("Number of received CurrentDemandRes messages: " &
          toString<integer>(v_loopCounter + 1));
    log("-------------------------------------------------------------");


    // MeteringReceipt
    if (verdict == pass) {
      if(vc_receiptRequired and PICS_CMN_CMN_IdentificationMode == pnC){
          verdict := f_SECC_DC_TB_VTB_MeteringReceipt_001(inconc); }
    } else { return verdict; }

    if (verdict == pass) {
      sleep(par_SECC_chargingLoop_pause);
      v_loopCounter := v_loopCounter + 1;
    }
}
if (vc_EVSENotification == reNegotiation or
    v_loopCounter == PICS_CMN_CMN_RenegotiationLoopIndication and
    PICS_CMN_CMN_Renegotiation){

    vc_EVTargetCurrent.value_ := 0;
    // CurrentDemand
    if (verdict == pass) {
       if(true){ verdict := f_SECC_DC_TB_VTB_CurrentDemand_001(inconc); }}
    else { return verdict;}

    v_loopCounter := v_loopCounter + 1;
    log("-------------------------------------------------------------");
    log("Number of received CurrentDemandRes messages: " &
          toString<integer>(v_loopCounter));
    log("-------------------------------------------------------------");

    // MeteringReceipt
    if (verdict == pass) {
       if(vc_receiptRequired and PICS_CMN_CMN_IdentificationMode == pnC){
           verdict := f_SECC_DC_TB_VTB_MeteringReceipt_001(inconc); }
    } else { return verdict; }
    if (verdict == pass) {
        sleep(par_SECC_chargingLoop_pause);
    }
    if(verdict == pass) {
      vc_EVTargetCurrent := { multiplier := 0, unit := a, value_ := 1   };
      vc_ChargeProgress := renegotiate;}
    else { return verdict; }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
        }
    }
    return verdict;
}

function f_SECC_DC_PR_PowerDeliveryRenegotiation_001(out HAL_61851_Listener
                                                     v_HAL_61851_Listener)
                                                     runs on SECC_Tester return verdicttype {

   var verdicttype verdict;

   verdict := f_SECC_DC_PR_CurrentDemandOrMeteringReceiptRenegotiation_001(
              v_HAL_61851_Listener);

   // f_SECC_PowerDeliveryStopDC Behavior
   if (verdict == pass) {
        verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(renegotiate, v_HAL_61851_Listener,
                                                       false, true, inconc);
   }
   return verdict;
}

function f_SECC_DC_PR_ChargeParameterDiscoveryRenegotiation_001(out HAL_61851_Listener
                                                                v_HAL_61851_Listener)
                                                                runs on SECC_Tester
                                                                return verdicttype {

   var verdicttype verdict;
   verdict := f_SECC_DC_PR_PowerDeliveryRenegotiation_001(v_HAL_61851_Listener);

   // f_SECC_ChargeParameterDiscovery Behavior
   if (verdict == pass) {
      vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 50 };
      if(PIXIT_SECC_CMN_SalesTariff == unknown) {
          verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
      } else {verdict := f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc); }
      vc_EVTargetCurrent := { multiplier := 0, unit := a, value_ := 1   };
   }

   // CableCheck
   if (verdict == pass) {
       verdict := f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, inconc);
   } else { return verdict; }

   // PreCharge
   if (verdict == pass) {
       vc_EVTargetVoltage := par_EVBatteryVoltage;
       verdict := f_SECC_DC_TB_VTB_PreCharge_001(inconc);
   } else { return verdict; }

   return verdict;
}

function f_SECC_DC_PR_PowerDeliveryStop_001(out HAL_61851_Listener v_HAL_61851_Listener,
                                            in boolean v_setState) runs on SECC_Tester
                                            return verdicttype {

   var verdicttype verdict;

   verdict := f_SECC_DC_PR_CurrentDemandOrMeteringReceiptStop_001(
                 v_HAL_61851_Listener);

   // SECC_PowerDeliveryStopDC Behavior
   if (verdict == pass) {
         vc_DC_EVStatus.eVReady := false;
         verdict := f_SECC_DC_TB_VTB_PowerDelivery_001(stop_, v_HAL_61851_Listener,
                                                        true, v_setState, inconc);
   }
   return verdict;
}

function f_SECC_DC_PR_WeldingDetection_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                           runs on SECC_Tester return verdicttype {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   var verdicttype verdict;

   verdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);

   // SECC_WeldingDetection Behavior
   if (verdict == pass) {
         verdict := f_SECC_DC_TB_VTB_WeldingDetection_001(inconc);
   }
   return verdict;
}

function f_SECC_DC_PR_SessionStop_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                      runs on SECC_Tester return verdicttype {

   var verdicttype verdict;

   if(PICS_SECC_DC_WeldingDetection) {
         verdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);
   }
   else {
      verdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);
   }

   // SECC_WeldingDetection Behavior
   if (verdict == pass) {
         verdict := f_SECC_CMN_TB_VTB_SessionStop_001(pause, v_HAL_61851_Listener,
                                                      inconc);
   }
   return verdict;
}

function f_SECC_DC_PR_SessionStop_002(out HAL_61851_Listener v_HAL_61851_Listener)
                                      runs on SECC_Tester return verdicttype {

   var verdicttype verdict;

   if(PICS_SECC_DC_WeldingDetection) {
      verdict := f_SECC_DC_PR_WeldingDetection_001(v_HAL_61851_Listener);
   }
   else {
      verdict := f_SECC_DC_PR_PowerDeliveryStop_001(v_HAL_61851_Listener, true);
   }

   // SECC_WeldingDetection Behavior
   if (verdict == pass) {
         verdict := f_SECC_CMN_TB_VTB_SessionStop_001(terminate, v_HAL_61851_Listener,
                                                      inconc);
   }
   return verdict;
}

// Paused Session

function f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(out HAL_61851_Listener
                                                             v_HAL_61851_Listener)
                                                             runs on SECC_Tester
                                                             return verdicttype {

    var verdicttype verdict := pass;

    v_HAL_61851_Listener.stop;
    v_HAL_61851_Listener.start(f_SECC_HAL_61851_Listener(true, false, e_StateB));

    if(PICS_CMN_CMN_ChargingMode == aC){
        f_SECC_changeValidDutyCycleRange(par_AC_validNominalDutyCycleLowerBound,
                                         par_AC_validNominalDutyCycleUpperBound);
        vc_validDutyCycleLowerBound2 := par_AC_validNominalDutyCycleLowerBound;
        vc_validDutyCycleUpperBound2 := par_AC_validNominalDutyCycleUpperBound;
    } else {
        vc_validDutyCycleLowerBound2 := par_validDutyCycleLowerBound;
        vc_validDutyCycleUpperBound2 := par_validDutyCycleUpperBound;
    }

    if(PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        // B toggle
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        tc_TP_EV_vald_state_duration.start(par_TP_EV_vald_state_duration);
        f_SECC_setState(e_StateB,v_HAL_61851_Listener);
        tc_TP_EV_vald_state_duration.timeout;
        // C toggle
        tc_TP_EV_vald_state_duration.start(par_TP_EV_vald_state_duration);
        f_SECC_changeValidStateCondition(e_StateC);
        f_SECC_setState(e_StateC,v_HAL_61851_Listener);
        tc_TP_EV_vald_state_duration.timeout;
        // B toggle
        tc_TP_EV_vald_state_duration.start(par_TP_EV_vald_state_duration);
        f_SECC_changeValidStateCondition(e_StateB);
        f_SECC_setState(e_StateB,v_HAL_61851_Listener);
    } else {
        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_confirmDutyCycle(v_HAL_61851_Listener,
                                par_CMN_waitForSUTInitiatedWakeUp,
                                fail);

        if (getverdict == pass) {
            setverdict(pass, "The SUT did initiate a wake-up " &
                             "within 'par_CMN_waitForSUTInitiatedWakeUp' " &
                             "after sleeping phase.");
        } else {
            setverdict(fail, "The SUT did not initiate a wake-up " &
                             "within 'par_CMN_waitForSUTInitiatedWakeUp' " &
                             "after sleeping phase.");
        }
    }
    if(getverdict == pass and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        tc_T_conn_max_comm.start(par_T_conn_max_comm);
    }
    f_SECC_getPLCLinkEstablishmentAfterSleepMode(pass, fail, fail);
    unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);

    if(getverdict == pass and PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        f_SECC_setIsConfirmationFlagDC(true);
        verdict := f_SECC_confirmDutyCycle(v_HAL_61851_Listener,
                   (par_T_conn_max_comm - tc_T_conn_max_comm.read),
                   fail);
    }

    if(getverdict == pass) {
        tc_V2G_EVCC_CommunicationSetup_Timer.start;

        //  SECC_Binding Behavior
        var Security_TYPE v_security := cc_hexTls;
        if(PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_SECC_CMN_TLS == false) {
            v_security := cc_hexTcp;
        }
        verdict := f_SECC_CMN_TB_VTB_SDP_001(v_security, inconc);
        if(vc_Security == cc_hexTcp) {
            vc_v2g_tcp_tls_parameter := md_CMN_CMN_TcpParameter_001(vc_V2G_Port_IpAddress,
                                                                    vc_V2G_Port_PortNumber,
                                                                    omit);
        } else if(vc_Security == cc_hexTls) {

            vc_v2g_tcp_tls_parameter := md_CMN_CMN_TlsParameter_002(
                                        vc_V2G_Port_IpAddress,
                                        vc_V2G_Port_PortNumber,
                                        omit, vc_cipherSuites,
                                        omit, vc_v2gRootCert);
        } else {log("Invalid security type was chosen.")};

        // SECC_SupportedAppProtocol Behavior
        if (verdict == pass) {
            f_SECC_CMN_PR_InitConfiguration_002(system, pass, inconc);
            verdict := getverdict;
            if (verdict == pass) {
                verdict := f_SECC_CMN_TB_VTB_SupportedAppProtocol_001(inconc);
                //clear SDP Port
                pt_V2G_UDP_SDP_Port.clear;
                vc_Default_SDP_Message := activate(a_SDP_Message());
            }
        }
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        return getverdict;
    }

    function f_SECC_CMN_PR_SessionSetupPausedSession_001(out HAL_61851_Listener
                                                         v_HAL_61851_Listener)
                                                         runs on SECC_Tester
                                                         return verdicttype {

        var verdicttype verdict;

        verdict := f_SECC_CMN_PR_SupportedAppProtocolPausedSession_001(v_HAL_61851_Listener);

        // SECC_SessionSetup Behavior
        if (verdict == pass) {
            verdict := f_SECC_CMN_TB_VTB_SessionSetup_005(inconc);
        }
        return verdict;
    }

    function f_SECC_CMN_PR_AuthorizationPausedSession_001(out HAL_61851_Listener
                                                          v_HAL_61851_Listener)
                                                          runs on SECC_Tester
                                                          return verdicttype {

        var verdicttype verdict;

        verdict := f_SECC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);

        // SECC_ServiceDiscovery Behavior
        if (verdict == pass) {
             verdict := f_SECC_CMN_TB_VTB_ServiceDiscovery_005(inconc);
        }
        // SECC_ServiceDetail Behavior
        if (verdict == pass and (ispresent(vc_serviceList.service_list))) {
             verdict := f_SECC_CMN_TB_VTB_ServiceDetail_001(inconc);
        }
        // SECC_PaymentServiceSelection Behavior
        if (verdict == pass) {
             verdict := f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(inconc);
        }
        if (verdict == pass) {
            if(PIXIT_SECC_CMN_CertificateHandling == certInstall and
               PICS_CMN_CMN_IdentificationMode == pnC) {
               verdict := f_SECC_CMN_TB_VTB_CertificateInstallation_001(inconc);
            }
        }
        if (verdict == pass) {
            if(PIXIT_SECC_CMN_CertificateHandling == certUpdate and
               PICS_CMN_CMN_IdentificationMode == pnC) {
               verdict := f_SECC_CMN_TB_VTB_CertificateUpdate_001(inconc);
            }
        }
        // SECC_PaymentDetails Behavior
        if (verdict == pass) {
            if(PICS_CMN_CMN_IdentificationMode == pnC) {
                verdict := f_SECC_CMN_TB_VTB_PaymentDetails_001(inconc);
            }
        }
        // SECC_Authorization Behavior
        if (verdict == pass) {
            if(PICS_CMN_CMN_IdentificationMode == eIM){
                verdict := f_SECC_CMN_TB_VTB_Authorization_001(inconc);
            }
            if(PICS_CMN_CMN_IdentificationMode == pnC){
                verdict := f_SECC_CMN_TB_VTB_Authorization_002(inconc);
            }
        }
        return verdict;
    }


}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

F.3.2 EVCC functions

```
module PreConditions_EVCC_15118_2 {

    import from Services_HAL_61851_1 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Pics_15118_2 all;
    import from Pixit_15118_2 all;
    import from Pics_15118 all;
    import from ComponentsAndPorts all;
    import from DataStructure_HAL_61851 all;
    import from DataStructure_SDP all;
    import from Configurations_15118_2 all;
    import from Timer_15118_2 all;
    import from LibFunctions_15118_2 all;
    import from TestBehavior_EVCC_SDP all;
    import from TestBehavior_EVCC_SupportedAppProtocol all;
    import from TestBehavior_EVCC_SessionSetup all;
    import from TestBehavior_EVCC_ServiceDetail all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from TestBehavior_EVCC_ServiceDetailAndPaymentSelection all;
    import from TestBehavior_EVCC_CertificateHandlingAndPaymentDetails all;
    import from TestBehavior_EVCC_CertificateInstallation all;
    import from TestBehavior_EVCC_CertificateUpdate all;
    import from TestBehavior_EVCC_Authorization all;
    import from TestBehavior_EVCC_ChargeParameterDiscovery all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from TestBehavior_EVCC_ChargingStatus all;
    import from TestBehavior_EVCC_ChargingLoopAcOrPowerDelivery all;
    import from TestBehavior_EVCC_MeteringReceipt all;
    import from TestBehavior_EVCC_CableCheck all;
    import from TestBehavior_EVCC_PreCharge all;
    import from TestBehavior_EVCC_PreChargeOrPowerDelivery all;
    import from TestBehavior_EVCC_CurrentDemand all;
    import from TestBehavior_EVCC_ChargingLoopDcOrPowerDelivery all;
    import from TestBehavior_EVCC_SessionStop all;
    import from TestBehavior_EVCC_WeldingDetection all;
    import from TestBehavior_EVCC_WeldingDetectionOrSessionStop all;
    import from TestBehavior_EVCC_CmSetKey all;
    import from Services_Security all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Templates_CMN_Security all;
    import from urn_iso_15118_2_2013_X509Def all;

    import from Pixit_15118 all;
    import from Timer_15118 all;
    import from PreConditions_EVCC_15118_3 { function f_EVCC_CMN_PR_CmAmpMap_001};
    import from Services_EVCC_PLCLinkStatus all;
    import from Timer_15118_3 all;
    import from TTlibrary_Logging all;

    //    PreConditions for EVCC-TestCases


    function f_EVCC_CMN_PR_GetSecurityData_001(out HAL_61851_Listener
                                        v_HAL_61851_Listener)
                                        runs on EVCC_Tester
                                        return verdicttype {

        var verdicttype verdict :=pass;

        if(PICS_CMN_CMN_IdentificationMode == pnC or PIXIT_EVCC_CMN_TLS == tls) {

            if(PIXIT_CMN_CMN_PkiName == PKI_6) {

                if(not PIXIT_EVCC_CMN_TLS == tls) {
                    setverdict(inconc, "Invalid PKI was selected. PKI-6 can only be used for " &
                                       "use case EIM with TLS support. Please check PICS and " &
                                       "PIXIT configuration.");
                } else {

                    vc_leafCert := fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                                      cc_crt_pe_leaf_valid);
                }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        } else {

            vc_leafCert := fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                              cc_crt_evse_leaf_valid);
            vc_subCert2 := fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                              cc_crt_cpo_sub2_valid);
            vc_subCert1 := fx_loadCertificate(PIXIT_CMN_CMN_PkiName,
                                              cc_crt_cpo_sub1_valid);

            vc_privateKey := fx_loadPrivateKey(cc_crt_mo_sub2_valid);

            if(PICS_CMN_CMN_IdentificationMode == pnC) {
               fx_setIdReferenceSA(vc_contractCertIdName,
                                   vc_emaid_idName,
                                   vc_encryptedPrivateKeyIdName,
                                   vc_dhPublicKeyIdName,
                                   vc_salesTariffIdName);
            }
        }
    }

    return verdict;
}

function f_EVCC_CMN_PR_PLCLinkStatusCombinedTesting_001(out HAL_61851_Listener
                                                        v_HAL_61851_Listener)
                                                        runs on EVCC_Tester
                                                        return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_GetSecurityData_001();

    if(PICS_CMN_CMN_CombinedTesting) {
        verdict := f_EVCC_CMN_PR_CmAmpMap_001(v_HAL_61851_Listener);
    }
    return verdict;
}

function f_EVCC_CMN_PR_DisableControlPilot_001(out HAL_61851_Listener
                                               v_HAL_61851_Listener)
                                               runs on EVCC_Tester
                                               return verdicttype {
    var float v_UnplugTimeRnd := 0.015;
    var verdicttype verdict := f_EVCC_CMN_PR_PLCLinkStatusCombinedTesting_001(
                                                        v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting and verdict == pass){
        verdict := f_EVCC_setPwmMode(e_OpenCircuit);
        sleep(v_UnplugTimeRnd);
        verdict := f_EVCC_setProximityOpenCircuit(true);
    }
    return verdict;
}

function f_EVCC_CMN_PR_DutyCycle_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                     runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_DisableControlPilot_001(
                                               v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting and verdict == pass){
        if(PICS_CMN_CMN_ChargingMode == aC){
            verdict := f_EVCC_setDutyCycle(vc_DutyCycle);
        }
        else {
            verdict := f_EVCC_setDutyCycle(5.0);
        }
    }
    return verdict;
}

function f_EVCC_CMN_PR_SetProximityPilot_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                             runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_DutyCycle_001(v_HAL_61851_Listener);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    if(not PICS_CMN_CMN_CombinedTesting and verdict == pass){

        if(PICS_CMN_CMN_PlugType == type1) {
            verdict := f_EVCC_setProximityResistance(cc_evse_proximity_type1);
        } else {
            if(PICS_CMN_CMN_ChargingMode == aC){
                var integer v_proximity_type2_AC;
                if(PICS_CMN_AC_CableCapability == capability13A) {
                    v_proximity_type2_AC := cc_proximity_type2_AC_13A;
                } else if(PICS_CMN_AC_CableCapability == capability20A) {
                    v_proximity_type2_AC := cc_proximity_type2_AC_20A;
                } else if(PICS_CMN_AC_CableCapability == capability32A) {
                    v_proximity_type2_AC := cc_proximity_type2_AC_32A;
                } else {
                    v_proximity_type2_AC := cc_proximity_type2_AC_63A;
                }
                verdict := f_EVCC_setProximityResistance(v_proximity_type2_AC);
            } else {
                verdict := f_EVCC_setProximityResistance(cc_evse_proximity_type2_DC);
            }
        }

        v_HAL_61851_Listener.stop;
        v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(true));

    }
    return verdict;
}

function f_EVCC_CMN_PR_EnergyTransferModeConfiguration_001(out HAL_61851_Listener
                                                           v_HAL_61851_Listener)
                                                           runs on EVCC_Tester
                                                           return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_SetProximityPilot_001(v_HAL_61851_Listener);

    if(PICS_CMN_CMN_ChargingMode == aC){
        vc_requestedEnergyTransferMode := vc_requestedEnergyTransferModeAC;
    } else {
        vc_requestedEnergyTransferMode := vc_requestedEnergyTransferModeDC;
    }
    return verdict;
}

function f_EVCC_CMN_PR_EnableControlPilot_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                              runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_EnergyTransferModeConfiguration_001(
                                                            v_HAL_61851_Listener);

    if(not PICS_CMN_CMN_CombinedTesting and verdict == pass){
        f_EVCC_changeValidStateCondition(e_StateE,e_StateAorValid);
        sleep(par_CMN_waitForNextHAL);
        verdict := f_EVCC_setPwmMode(e_OscOn);
    }
    return verdict;
}

function f_EVCC_CMN_PR_StateB_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                  runs on EVCC_Tester return verdicttype {

  var verdicttype verdict := f_EVCC_CMN_PR_EnableControlPilot_001(v_HAL_61851_Listener);

  if(not PICS_CMN_CMN_CombinedTesting and verdict == pass){
     timer t_pluginDetection := par_EVCC_waitForPlugin;
     verdict := f_EVCC_confirmState(e_StateB, v_HAL_61851_Listener,
                                    t_pluginDetection, inconc, true);
  }
  tc_V2G_SECC_CommunicationSetup_Timer.start;
  verdict := getverdict;
  return verdict;
}

function f_EVCC_CMN_PR_SDP_001(out HAL_61851_Listener v_HAL_61851_Listener)
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_StateB_001(v_HAL_61851_Listener);

    if(vc_unmapSlac) {
        f_EVCC_PO_ShutdownConfiguration_002(system);
    }

    //  EVCC_Binding Behavior
    if (verdict == pass) {
         var template Security_TYPE v_security := ?;
         verdict := f_EVCC_CMN_TB_VTB_SDP_001(v_security, inconc);
         if(PICS_CMN_CMN_IdentificationMode == pnC) {
            if(vc_Security != cc_hexTls){
                setverdict(inconc, "SUT did not request the correct transmission security. " &
                                   "TLS was expected.");
            }
         }
         if(PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == tls ) {
            if(vc_Security != cc_hexTls){
               setverdict(inconc, "SUT did not request the correct transmission security. " &
                                     "TLS was expected.");
            }
         }
         if(PICS_CMN_CMN_IdentificationMode == eIM and PIXIT_EVCC_CMN_TLS == none_ ) {
            if(vc_Security != cc_hexTcp){
              setverdict(inconc, "SUT did not request the correct transmission security. " &
                                   "TCP was expected.");
            }
         }
         if(getverdict == pass) {
             if(vc_Security == cc_hexTcp) {
                vc_v2g_tcp_tls_parameter := md_CMN_CMN_TcpParameter_001(
                                                    vc_sut_ipAddress.ipAddress,
                                                    vc_V2G_Port_PortNumber,
                                                    vc_Security);
             } else if(vc_Security == cc_hexTls) {

                vc_v2g_tcp_tls_parameter := md_CMN_CMN_TlsParameter_001(
                                                    vc_sut_ipAddress.ipAddress,
                                                    vc_V2G_Port_PortNumber,
                                                    vc_Security, vc_cipherSuites,
                                                    vc_enabledProtocol,
                                                    md_CertificateChainType_002(
                                                    vc_leafCert, vc_subCert2,
                                                    vc_subCert1));
             } else {log("Invalid security type was chosen.")};
         }
    }
    return getverdict;
}

function f_EVCC_CMN_PR_SupportedAppProtocol_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                                runs on EVCC_Tester return verdicttype {
    var verdicttype verdict;

        verdict := f_EVCC_CMN_PR_SDP_001(v_HAL_61851_Listener);
    if (verdict == pass) {
        f_EVCC_CMN_PR_InitConfiguration_002(system, inconc);
        //clear SDP Port
        pt_V2G_UDP_SDP_Port.clear;
        verdict := f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001(inconc);
        if(verdict == pass) {
           vc_TcpPortIsMapped := true;
        }
        vc_Default_SDP_Message := activate(a_SDP_Message());
    }
    return verdict;
}

function f_EVCC_CMN_PR_SessionSetup_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                        runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_SupportedAppProtocol_001(v_HAL_61851_Listener);
    if (verdict == pass) {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        verdict := f_EVCC_CMN_TB_VTB_SessionSetup_001(inconc);
    }
    return verdict;
}

function f_EVCC_CMN_PR_ServiceDiscovery_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                            runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_SessionSetup_001(v_HAL_61851_Listener);
    if (verdict == pass) {
        verdict := f_EVCC_CMN_TB_VTB_ServiceDiscovery_001(inconc);
    }
    return verdict;
}

function f_EVCC_CMN_PR_ServiceDetail_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                         runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);
    if (verdict == pass) {
        if (PICS_CMN_CMN_IdentificationMode == pnC) {
            verdict := f_EVCC_CMN_TB_VTB_ServiceDetail_001(inconc); }
        if (PICS_CMN_CMN_IdentificationMode == eIM) {
            verdict := f_EVCC_CMN_TB_VTB_ServiceDetail_002(inconc); }
    }
    return verdict;
}

function f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(out HAL_61851_Listener
                                                            v_HAL_61851_Listener)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_ServiceDiscovery_001(v_HAL_61851_Listener);
    vc_ServiceId := 1;
    if (verdict == pass) {
        if (PICS_CMN_CMN_IdentificationMode == pnC) {
            verdict := f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_001(inconc); }
        if (PICS_CMN_CMN_IdentificationMode == eIM) {
            verdict := f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_002(inconc); }
    }
    return verdict;
}

function f_EVCC_CMN_PR_CertificateInstallation_001(out HAL_61851_Listener
                                                   v_HAL_61851_Listener)
                                                   runs on EVCC_Tester
                                                   return verdicttype {

    var verdicttype verdict;

        verdict := f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);
    if (verdict == pass) {
        verdict := f_EVCC_CMN_TB_VTB_CertificateInstallation_001(inconc);
    }
    return verdict;
}

function f_EVCC_CMN_PR_CertificateUpdate_001(out HAL_61851_Listener
                                             v_HAL_61851_Listener)
                                             runs on EVCC_Tester return verdicttype {

    var verdicttype verdict;

        verdict := f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);
    if (verdict == pass) {
        verdict := f_EVCC_CMN_TB_VTB_CertificateUpdate_001(inconc);
    }
    return verdict;
}

function f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(out HAL_61851_Listener
                                                                v_HAL_61851_Listener)
                                                                runs on EVCC_Tester
                                                                return verdicttype {
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var verdicttype verdict;

        verdict := f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);
    if (verdict == pass) {
        verdict := f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_001(inconc);
    }
    return verdict;
}

function f_EVCC_CMN_PR_Authorization_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                         runs on EVCC_Tester return verdicttype {

    var verdicttype verdict;
    if (PICS_CMN_CMN_IdentificationMode == pnC) { verdict :=
        f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener); }
    if (PICS_CMN_CMN_IdentificationMode == eIM) {
            verdict := f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);
    }
    if (verdict == pass) {
        if (PICS_CMN_CMN_IdentificationMode == pnC) {
            verdict := f_EVCC_CMN_TB_VTB_Authorization_008(inconc);
        } else if (PICS_CMN_CMN_IdentificationMode == eIM) {
            verdict := f_EVCC_CMN_TB_VTB_Authorization_002(inconc);
        }
    }
    return verdict;
}

function f_EVCC_CMN_PR_Authorization_002(out HAL_61851_Listener v_HAL_61851_Listener)
                                         runs on EVCC_Tester return verdicttype {

    var verdicttype verdict;
    if (PICS_CMN_CMN_IdentificationMode == pnC) { verdict :=
        f_EVCC_CMN_PR_CertificateHandlingAndPaymentDetails_001(v_HAL_61851_Listener); }
    if (PICS_CMN_CMN_IdentificationMode == eIM) {
            verdict := f_EVCC_CMN_PR_ServiceDetailAndPaymentSelection_001(v_HAL_61851_Listener);
    }
    if (verdict == pass) {
         verdict := f_EVCC_CMN_TB_VTB_Authorization_001(inconc);
    }
    return verdict;
}

function f_EVCC_CMN_PR_ChargeParameterDiscovery_001(out HAL_61851_Listener
                                                    v_HAL_61851_Listener)
                                                    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_Authorization_001(v_HAL_61851_Listener);
    if (verdict == pass) {
        verdict := f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_001(inconc);
    }
    return verdict;
}

function f_EVCC_AC_PR_PowerDeliveryStart_001(out HAL_61851_Listener
                                             v_HAL_61851_Listener)
                                             runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);
    if (verdict == pass) {
        verdict := f_EVCC_AC_TB_VTB_PowerDelivery_001(v_HAL_61851_Listener, inconc);
    }
    return verdict;
}

function f_EVCC_AC_PR_ChargingStatus_001(out HAL_61851_Listener
                                         v_HAL_61851_Listener)
                                         runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_AC_PR_PowerDeliveryStart_001(v_HAL_61851_Listener);
    if (verdict == pass and not vc_sleepModeWoCharging) {
        verdict := f_EVCC_AC_TB_VTB_ChargingStatus_001(v_HAL_61851_Listener, inconc);
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    return verdict;
}

function f_EVCC_AC_PR_MeteringReceipt_001(out HAL_61851_Listener
                                          v_HAL_61851_Listener)
                                          runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);
    if (verdict == pass and not vc_sleepModeWoCharging) {
        verdict := f_EVCC_AC_TB_VTB_MeteringReceipt_001(inconc);
    }
    return verdict;
}

function f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(out HAL_61851_Listener
                                                        v_HAL_61851_Listener)
                                                        runs on EVCC_Tester
                                                        return verdicttype {

    var verdicttype verdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);
    if (verdict == pass and not vc_sleepModeWoCharging) {
        if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
              verdict :=
              f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_002(v_HAL_61851_Listener,
                                                                 inconc);
        }
        else {
            verdict :=
            f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener,
                                                               inconc);
        }
    }
    return verdict;
}

function f_EVCC_AC_PR_SessionStop_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                      runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);
    if (verdict == pass) {
      verdict := f_EVCC_AC_TB_VTB_SessionStop_002(v_HAL_61851_Listener, inconc);
    }
    return verdict;
}

function f_EVCC_AC_PR_SessionStop_002(out HAL_61851_Listener v_HAL_61851_Listener)
                                      runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);
    if (verdict == pass) {
      verdict := f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, inconc, pass, inconc);
    }
    return verdict;
}

function f_EVCC_AC_PR_SessionStop_003(out HAL_61851_Listener v_HAL_61851_Listener)
                                      runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_AC_PR_ChargingLoopAcOrPowerDelivery_001(v_HAL_61851_Listener);
    if (verdict == pass) {
      verdict := f_EVCC_AC_TB_VTB_SessionStop_001(v_HAL_61851_Listener, pass, inconc, inconc);
    }
    return verdict;
}

// Renegotiation
function f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(out HAL_61851_Listener
                                                                     v_HAL_61851_Listener)
                                                                     runs on EVCC_Tester
                                                                     return verdicttype {

    var verdicttype verdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
        verdict := f_EVCC_AC_PR_MeteringReceipt_001(v_HAL_61851_Listener);
    }
    else {verdict := f_EVCC_AC_PR_ChargingStatus_001(v_HAL_61851_Listener);}

    if (verdict == pass){
        if (PICS_CMN_CMN_Renegotiation and PIXIT_CMN_CMN_Renegotiation == unknown) {
            if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
                verdict :=
                f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_006(v_HAL_61851_Listener,
                                                                   inconc);
            }
            else {verdict :=
                  f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_005(v_HAL_61851_Listener,
                                                                     inconc);}
        }
        else if (not PICS_CMN_CMN_Renegotiation and
                 PIXIT_CMN_CMN_Renegotiation == renegotiation) {
            if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
                verdict :=
                f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_010(v_HAL_61851_Listener,
                                                                   inconc);
            }
            else {verdict :=
                  f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_009(v_HAL_61851_Listener,
                                                                     inconc);}
        }
    }
    return verdict;
}

// Renegotiation
function f_EVCC_AC_PR_ChargeParameterDiscoveryRenegotiation_001(out HAL_61851_Listener
                                                                v_HAL_61851_Listener)
                                                                runs on EVCC_Tester
                                                                return verdicttype {

    var verdicttype verdict;
    verdict := f_EVCC_AC_PR_ChargingLoopAcOrPowerDeliveryRenegotiation_001(
                                                    v_HAL_61851_Listener);

    if (verdict == pass){
        verdict := f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_003(inconc);
    }
    return verdict;
}

//  PreConditions for DC-TestCases

function f_EVCC_DC_PR_CableCheck_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                     runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);
    if (verdict == pass){
        verdict := f_EVCC_DC_TB_VTB_CableCheck_007(v_HAL_61851_Listener, inconc);
    }
    return verdict;
}

function f_EVCC_DC_PR_CableCheck_002(out HAL_61851_Listener v_HAL_61851_Listener)
                                     runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_CMN_PR_ChargeParameterDiscovery_001(v_HAL_61851_Listener);
    if (verdict == pass){
        verdict := f_EVCC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, inconc);
    }
    return verdict;
}

function f_EVCC_DC_PR_PreCharge_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                    runs on EVCC_Tester return verdicttype {
    var verdicttype verdict;
    verdict := f_EVCC_DC_PR_CableCheck_001(v_HAL_61851_Listener);
    if (verdict == pass){
        verdict := f_EVCC_DC_TB_VTB_PreCharge_001(inconc);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }
    return verdict;
}

function f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(out HAL_61851_Listener
                                                    v_HAL_61851_Listener)
                                                    runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_DC_PR_PreCharge_001(v_HAL_61851_Listener);
    if (verdict == pass){
        verdict := f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001(inconc);
    }
    return verdict;
}

function f_EVCC_DC_PR_CurrentDemand_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                         runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_DC_PR_PreChargeOrPowerDelivery_001(v_HAL_61851_Listener);
    if (verdict == pass and not vc_sleepModeWoCharging){
        verdict := f_EVCC_DC_TB_VTB_CurrentDemand_001(inconc);
    }
    return verdict;
}

function f_EVCC_DC_PR_MeteringReceipt_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                           runs on EVCC_Tester return verdicttype {

    var verdicttype verdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);
    if ( verdict == pass and not vc_sleepModeWoCharging) {
        verdict := f_EVCC_DC_TB_VTB_MeteringReceipt_001(inconc);
    }
    return verdict;
}

function f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(out HAL_61851_Listener
                                                         v_HAL_61851_Listener)
                                                         runs on EVCC_Tester
                                                         return verdicttype {

    var verdicttype verdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);

    if (verdict == pass and not vc_sleepModeWoCharging){
        if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
            verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_002(v_HAL_61851_Listener,
                                                                          inconc);
        }
        else {verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener,
                                                                            inconc);;}
    }
    return verdict;
}

function f_EVCC_DC_PR_WeldingDetection_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                            runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);
    if (verdict == pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            verdict := f_EVCC_DC_TB_VTB_WeldingDetection_001(v_HAL_61851_Listener, inconc);
        }
    }
    return verdict;
}

function f_EVCC_DC_PR_WeldingDetectionOrSessionStop_001(out HAL_61851_Listener
                                                         v_HAL_61851_Listener)
                                                         runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    if (verdict == pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            verdict := f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_003(v_HAL_61851_Listener,
                                                                          inconc);
        }
    }
    return verdict;
}

function f_EVCC_DC_PR_WeldingDetectionOrSessionStop_002(out HAL_61851_Listener
                                                        v_HAL_61851_Listener)
                                                        runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);
    if (verdict == pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            verdict := f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                          inconc, pass, inconc);
        }
    }
    return verdict;
}

function f_EVCC_DC_PR_WeldingDetectionOrSessionStop_003(out HAL_61851_Listener
                                                        v_HAL_61851_Listener)
                                                        runs on EVCC_Tester return verdicttype {

    var verdicttype verdict :=
    f_EVCC_DC_PR_ChargingLoopDcOrPowerDelivery_001(v_HAL_61851_Listener);
    if (verdict == pass){
        log("Send PowerDeliveryRes message with delay of 0.5s");
        sleep(0.5);
        if (getverdict == pass){
            verdict := f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(v_HAL_61851_Listener,
                                                                          pass, inconc, inconc);
        }
    }
    return verdict;
}

// Renegotiation
function f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(out HAL_61851_Listener
                                                                     v_HAL_61851_Listener)
                                                                     runs on EVCC_Tester
                                                                     return verdicttype {

    var verdicttype verdict;
    if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
        verdict := f_EVCC_DC_PR_MeteringReceipt_001(v_HAL_61851_Listener);
    }
    else {verdict := f_EVCC_DC_PR_CurrentDemand_001(v_HAL_61851_Listener);}

    if (verdict == pass){
        if (PICS_CMN_CMN_Renegotiation and PIXIT_CMN_CMN_Renegotiation == unknown) {
            if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
                verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_006(inconc);
            }
            else {verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_005(inconc);}
        }
        else if (not PICS_CMN_CMN_Renegotiation and
                 PIXIT_CMN_CMN_Renegotiation == renegotiation) {
            if (PICS_CMN_CMN_IdentificationMode == pnC and PICS_EVCC_CMN_ReceiptRequired) {
                verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_010(inconc);
            }
            else {verdict := f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_009(inconc);}
        }
    }
    return verdict;
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
// Renegotiation
function f_EVCC_DC_PR_PreChargeRenegotiation_001(out HAL_61851_Listener
                                                 v_HAL_61851_Listener)
                                                 runs on EVCC_Tester
                                                 return verdicttype {
    var verdicttype verdict;
    verdict := f_EVCC_DC_PR_ChargingLoopDcOrPowerDeliveryRenegotiation_001(
                                                      v_HAL_61851_Listener);

    if (verdict == pass){
        verdict := f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_003(v_HAL_61851_Listener, inconc);
    }

    if (verdict == pass){
        verdict := f_EVCC_DC_TB_VTB_CableCheck_007(v_HAL_61851_Listener, inconc);
    }

    if (verdict == pass){
        verdict := f_EVCC_DC_TB_VTB_PreCharge_001(inconc);
    }
    return verdict;
}

// Sleep mode
function f_EVCC_CMN_PR_PausedSession_001(out HAL_61851_Listener
                                         v_HAL_61851_Listener)
                                         runs on EVCC_Tester
                                         return verdicttype {

    var verdicttype verdict := pass;
    var float v_time;
    timer t_statetimer;
    timer t_EVWakeUp;
    vc_V2G_Port_PortNumber := 0;

    v_HAL_61851_Listener.stop;
    v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(true));

     if(PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        f_EVCC_setDutyCycle(5.0);
        f_EVCC_setPwmMode(e_OscOn);
     } else {
        // BCB toggle sequence detection
        f_EVCC_changeValidStateCondition(e_StateB,e_StateC);
        t_EVWakeUp.start(par_CMN_waitForSUTInitiatedWakeUp);
        alt {
           [] a_EVCC_BCBToggleDetection(pt_HAL_61851_Internal_Port, e_StateB){

               t_statetimer.stop;
           }
           [] a_EVCC_BCBToggleDetection(pt_HAL_61851_Internal_Port, e_StateC){

               t_EVWakeUp.stop;
               f_EVCC_changeValidStateCondition(e_StateC,e_StateB);
               t_statetimer.start(par_T_vald_state_duration_max);
               repeat;
           }
           [] pt_HAL_61851_Internal_Port.receive {
               setverdict(inconc, "Received state has an invalid value.");
           }
           [] t_statetimer.timeout {
               setverdict(inconc, "The EVSE could not detect the corresponding " &
                                  "toggle value within " &
                                  "the maximal valid state duration.");
           }
           [] t_EVWakeUp.timeout {
               setverdict(fail, "The SUT did not initiate a wake-up " &
                                "within 'par_CMN_waitForSUTInitiatedWakeUp' " &
                                "after sleeping phase.");
           }
        }
        if(getverdict == pass) {
            f_EVCC_setDutyCycle(5.0);
            f_EVCC_setPwmMode(e_OscOn);
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     }

    f_EVCC_getPLCLinkEstablishmentAfterSleepMode(fail);
    unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);

    tc_V2G_SECC_CommunicationSetup_Timer.start;
    return getverdict;
}

function f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(out HAL_61851_Listener
                                                             v_HAL_61851_Listener)
                                                             runs on EVCC_Tester
                                                             return verdicttype {

    var verdicttype verdict := pass;
    var float v_time;
    timer t_statetimer;
    timer t_EVWakeUp;
    vc_V2G_Port_PortNumber := 0;

    v_HAL_61851_Listener.stop;
    v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(true));

    if(PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
        f_EVCC_setDutyCycle(5.0);
        f_EVCC_setPwmMode(e_OscOn);
    } else {
        // BCB toggle sequence detection
        f_EVCC_changeValidStateCondition(e_StateB,e_StateC);
        t_EVWakeUp.start(par_CMN_waitForSUTInitiatedWakeUp);
        alt {
           [] a_EVCC_BCBToggleDetection(pt_HAL_61851_Internal_Port, e_StateB){

               t_statetimer.stop;
           }
           [] a_EVCC_BCBToggleDetection(pt_HAL_61851_Internal_Port, e_StateC){

               t_EVWakeUp.stop;
               f_EVCC_changeValidStateCondition(e_StateC,e_StateB);
               t_statetimer.start(par_T_vald_state_duration_max);
               repeat;
           }
           [] pt_HAL_61851_Internal_Port.receive {
               setverdict(inconc, "Received state has an invalid value.");
           }
           [] t_statetimer.timeout {
               setverdict(fail, "The EVSE could not detect the corresponding " &
                                "toggle value within " &
                                "the maximal valid state duration.");
           }
           [] t_EVWakeUp.timeout {
               setverdict(fail, "The SUT did not initiate a wake-up " &
                                   "within 'par_CMN_waitForSUTInitiatedWakeUp' " &
                                   "after sleeping phase.");
           }
        }
        if(getverdict == pass) {
            f_EVCC_setDutyCycle(5.0);
            f_EVCC_setPwmMode(e_OscOn);
        }
    }
    f_EVCC_getPLCLinkEstablishmentAfterSleepMode(fail);
    unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);

    if(getverdict == pass) {
        tc_V2G_SECC_CommunicationSetup_Timer.start;
        //  EVCC_Binding Behavior
        if ( verdict == pass ) {
             var template Security_TYPE v_security := ?;
             verdict := f_EVCC_CMN_TB_VTB_SDP_001(v_security, inconc);
             if(PICS_CMN_CMN_IdentificationMode == pnC) {
                if(vc_Security != cc_hexTls){
                    setverdict(inconc, "SUT did not request the correct " &
                                       "transmission security.");
                }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                }
                if(PICS_CMN_CMN_IdentificationMode == eIM and
                   PIXIT_EVCC_CMN_TLS == none_ ) {
                   if(vc_Security != cc_hexTcp){
                      setverdict(inconc, "SUT did not request the correct " &
                                         "transmission security.");
                   }
                }
                if(getverdict == pass) {
                    if(vc_Security == cc_hexTcp) {
                       vc_v2g_tcp_tls_parameter := md_CMN_CMN_TcpParameter_001(
                                                   vc_sut_ipAddress.ipAddress,
                                                   vc_V2G_Port_PortNumber,
                                                   vc_Security);
                    } else if(vc_Security == cc_hexTls) {

                       vc_v2g_tcp_tls_parameter := md_CMN_CMN_TlsParameter_001(
                                                   vc_sut_ipAddress.ipAddress,
                                                   vc_V2G_Port_PortNumber,
                                                   vc_Security, vc_cipherSuites,
                                                   vc_enabledProtocol,
                                                   md_CertificateChainType_002(
                                                   vc_leafCert, vc_subCert2,
                                                   vc_subCert1));
                    } else {log("Invalid security type was chosen.")};
                }
            }
        }
        if (getverdict == pass) {
            f_EVCC_CMN_PR_InitConfiguration_002(system, inconc);
            //clear SDP Port
            pt_V2G_UDP_SDP_Port.clear;
            verdict := f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001(inconc);
            if(verdict == pass) {
               vc_TcpPortIsMapped := true;
            }
            vc_Default_SDP_Message := activate(a_SDP_Message());
        }
        return getverdict;
    }

    function f_EVCC_CMN_PR_SessionSetupPausedSession_001(out HAL_61851_Listener
                                                         v_HAL_61851_Listener)
                                                         runs on EVCC_Tester
                                                         return verdicttype {

        var verdicttype verdict;

        verdict := f_EVCC_CMN_PR_SupportedAppProtocolPausedSession_001(
                                                 v_HAL_61851_Listener);

        // SECC_SessionSetup Behavior
        if (verdict == pass) {
            verdict := f_EVCC_CMN_TB_VTB_SessionSetup_006(inconc);
        }
        if (verdict == pass) {
            verdict := f_EVCC_CMN_TB_VTB_ServiceDiscovery_006(inconc);
        }
        return verdict;
    }

    function f_EVCC_CMN_PR_AuthorizationPausedSession_001(out HAL_61851_Listener
                                                          v_HAL_61851_Listener)
                                                          runs on EVCC_Tester
                                                          return verdicttype {

        var verdicttype verdict;

        verdict := f_EVCC_CMN_PR_SessionSetupPausedSession_001(v_HAL_61851_Listener);

        // SECC_SessionSetup Behavior

        if (verdict == pass) {
            if (PICS_CMN_CMN_IdentificationMode == pnC) {
                verdict := f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(inconc); }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if (PICS_CMN_CMN_IdentificationMode == eIM) {
                verdict := f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(inconc); }
        }
        if (verdict == pass) {
            if (PICS_CMN_CMN_IdentificationMode == pnC) {
                verdict := f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_001(inconc); }
        }

        if (verdict == pass) {
            if (PICS_CMN_CMN_IdentificationMode == pnC) {
                verdict := f_EVCC_CMN_TB_VTB_Authorization_008(inconc);
            } else if (PICS_CMN_CMN_IdentificationMode == eIM) {
                verdict := f_EVCC_CMN_TB_VTB_Authorization_002(inconc);
            }
        }
        return verdict;
    }
}
```

F.4 Post-condition functions

F.4.1 SECC functions

```
module PostConditions_SECC_15118_2 {

    import from Pics_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from ComponentsAndPorts all;
    import from DataStructure_HAL_61851 all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 {modulepar par_CMN_waitForKeyReset;};
    import from DataStructure_V2G_TCP_TLS_Port_Control all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from TestBehavior_SECC_CommonBehavior all;

    import from Pics_15118 all;
    import from Pics_15118_2 all;
    import from Pixit_15118 all;
    import from Timer_15118 all;
    import from TTlibrary_Logging all;


    function f_SECC_CMN_PO_InitialState_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                            runs on SECC_Tester {

         var float v_waitForRes := 1.0;
         var float v_UnplugTimeRnd := 0.015;


         log("Wait for " & toString<float>(par_CMN_waitForPlugOut) & "s before " &
             "plug-out will be initiated.");
         sleep(par_CMN_waitForPlugOut);

              pt_HAL_61851_Internal_Port.clear;
             v_HAL_61851_Listener.stop;
             v_HAL_61851_Listener.start(f_SECC_HAL_61851_Listener(false, false, e_StateA));

         var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_internalResponseMessage;
         var boolean isClosed := false;
         var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;

         sleep(v_waitForRes);
         deactivate(vc_Default_IEC_61851_ListenerBehavior);
         deactivate(vc_Default_SDP_Message);

         if(pt_V2G_TCP_TLS_ALM_SECC_Port.checkstate("Mapped") and not(vc_isTcpPortClosed)){
             pt_V2G_TCP_TLS_ALM_SECC_Port.clear;
             pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_disconnectPort));
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            alt{
                [] pt_V2G_TCP_TLS_Port_Control.receive(md_CMN_CMN_tcpTlsStatusRes_001(
                                                    e_disconnectPort,?,?,*))
                                                    -> value v_internalResponseMessage {

                    isClosed := v_internalResponseMessage.result.portStatus.closed;
                    ackType := v_internalResponseMessage.ack;
                }
                [] pt_V2G_TCP_TLS_Port_Control.receive {
                    repeat;
                }
            }
        }

            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            f_SECC_changeValidStateCondition(e_StateB);
            f_SECC_setState(e_StateB, v_HAL_61851_Listener);
            sleep(par_CMN_waitForNextHAL);

            sleep(par_CMN_waitForNextHAL);
            f_SECC_changeValidStateCondition(e_StateA);
            f_SECC_setState(e_StateA,v_HAL_61851_Listener);
            sleep(v_UnplugTimeRnd);
            f_SECC_setProximityVoltage(false);
            f_SECC_setProximityOpenCircuit(true);




        sleep(par_CMN_waitForNextHAL);
           v_HAL_61851_Listener.stop;
        all timer.stop;

        log((par_SECC_waitForNextTC)," Sec timer started");
        sleep(par_SECC_waitForNextTC);
        log((par_SECC_waitForNextTC)," Sec timer stopped");


    }
}
```

F.4.2 EVCC functions

```
module PostConditions_EVCC_15118_2 {

    import from Pics_15118_2 all;

    import from Services_HAL_61851_1 all;
    import from ComponentsAndPorts all;
    import from DataStructure_HAL_61851 all;
    import from LibFunctions_15118_2 all;
    import from Pics_15118 all;
    import from Timer_15118_2 all;
    import from DataStructure_V2G_TCP_TLS_Port_Control all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Pixit_15118 all;
    import from Timer_15118 all;
    import from TTlibrary_Logging all;


    function f_EVCC_CMN_PO_InitialState_001(out HAL_61851_Listener v_HAL_61851_Listener)
                                        runs on EVCC_Tester {

        var float v_UnplugTimeRnd := 0.015;
        var float v_waitForPTCstop := 0.2;

            sleep(v_waitForPTCstop);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            v_HAL_61851_Listener.stop;
            v_HAL_61851_Listener.start(f_EVCC_HAL_61851_Listener(false));
            sleep(par_CMN_waitForNextHAL);
            pt_HAL_61851_Internal_Port.clear;
            pt_HAL_61851_Port.clear;
            f_EVCC_setPwmMode(e_PosVolt12);



        var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_internalResponseMessage;
        var boolean isClosed := false;
        var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;

        pt_V2G_TCP_TLS_ALM_EVCC_Port.clear;
        deactivate(vc_Default_IEC_61851_ListenerBehavior);
        deactivate(vc_Default_SDP_Message);

        if(vc_TcpPortIsMapped and not(vc_isTcpPortClosed)){
            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_disconnectPort));
           alt{
                [] pt_V2G_TCP_TLS_Port_Control.receive(md_CMN_CMN_tcpTlsStatusRes_001(
                                                    e_disconnectPort,?,?,*))
                                                    -> value v_internalResponseMessage {

                    isClosed := v_internalResponseMessage.result.portStatus.closed;
                    ackType := v_internalResponseMessage.ack;
                }
                [] pt_V2G_TCP_TLS_Port_Control.receive {
                    repeat;
                }
            }
        }


             sleep(par_CMN_waitForNextHAL);
             f_EVCC_setPwmMode(e_OpenCircuit);
             sleep(v_UnplugTimeRnd);
             f_EVCC_setProximityOpenCircuit(true);





         sleep(par_CMN_waitForNextHAL);
           v_HAL_61851_Listener.stop;
        all timer.stop;

        log((par_EVCC_waitForNextTC)," Sec timer started");
        sleep(par_EVCC_waitForNextTC);
        log((par_EVCC_waitForNextTC)," Sec timer stopped");


     }
}
```

F.5 Common behavior functions

F.5.1 SECC functions

```
module TestBehavior_SECC_CommonBehavior {

    import from DataStructure_HAL_61851 all;
    import from Templates_CMN_HAL61851 all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from ComponentsAndPorts all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from DataStructure_SDP all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  import from Services_SECC_PLCLinkStatus all;

  altstep a_SECC_Unexpected_Message_Content(verdicttype v_verdict, charstring v_message)
                                            runs on SECC_Tester {

    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive
     {
        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(v_verdict, v_message);
     }
  }

  altstep a_SECC_Unexpected_Message(verdicttype v_verdict, charstring v_message)
                                    runs on SECC_Tester {

    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive
     {
         setverdict(v_verdict, v_message);
     }
   }

  altstep a_SECC_Unexpected_V2G_Message(verdicttype v_verdict, charstring v_message,
                                        template BodyElement_group body)
                                        runs on SECC_Tester {

    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(?,
                                          md_CMN_BodyType_001(complement(body)))) {
        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(v_verdict, v_message);
     }
  }

 altstep a_SECC_Timeout(verdicttype v_verdict) runs on SECC_Tester {

    [] tc_V2G_EVCC_Msg_Timer.timeout {

        setverdict(v_verdict, "Nothing received. V2G_EVCC_Msg_Timer " &
                              "was triggered.");
     }
   }

altstep a_No_SDP_Message(charstring v_message, verdicttype v_verdict)
                         runs on SECC_Tester {

    [] pt_V2G_UDP_SDP_Port.receive {

        tc_EVCC_SDP_Timer.stop;
        setverdict(v_verdict, v_message);
     }
}

altstep a_SECC_SDP_Timeout(verdicttype v_verdict) runs on SECC_Tester {

    []  tc_EVCC_SDP_Timer.timeout {

        setverdict(v_verdict, "Nothing received. tc_EVCC_SDP_Timer " &
                              "was triggered.");
     }
}

  altstep a_SDP_Message() runs on SECC_Tester {

      [] pt_V2G_UDP_SDP_Port.receive(SDP_Message:?) {

          repeat;
      }
  }

altstep a_SECC_CommunicationSetup_Timeout(verdicttype v_verdict)
                                          runs on SECC_Tester {

   [] tc_V2G_EVCC_CommunicationSetup_Timer.timeout {

       setverdict(v_verdict, "Nothing received. V2G_EVCC_CommunicationSetup_Timer " &
                             "was triggered.");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      }
  }

  altstep a_SECC_TCPConnection_Status_Listener(verdicttype v_verdict, charstring v_message)
                                                  runs on SECC_Tester {

    [] pt_V2G_TCP_TLS_Port_Control.receive(md_CMN_CMN_tcpTlsStatusRes_001(
                                        e_disconnectPort, ?,?, *)) {

          setverdict(v_verdict,v_message);
          vc_isTcpPortClosed := true;
       }
  }

  altstep a_SECC_TCP_Connection_Termination_Timeout() runs on SECC_Tester {

      [] tc_CMN_TCP_Connection_Termination_Timer.timeout {
         log("TCP connection was >>NOT<< terminated by the SUT, " &
             "tc_CMN_TCP_Connection_Termination_Timer was triggered.");
      }
  }

   altstep a_SECC_CPOscillator_Shutdown_Timeout() runs on SECC_Tester {

     [] tc_SECC_CPOscillator_Shutdown_Timer.timeout {
         log("SECC oscillator was not shut down. " &
             "SECC_CPOscillator_Shutdown_Timer was triggered.");
     }
   }

  function isError() return boolean {

    if(getverdict == error) {
       return true;
    }
    return false;
  }

  function f_SECC_startSleepingPhase(in float v_time) runs on SECC_Tester
                                                        return verdicttype {

      timer t_sleep := v_time;
      deactivate(vc_Default_SDP_Message);

      t_sleep.start;
      alt {
          [] t_sleep.timeout {
             log("Sleeping phase was terminated by the Test System.")
          }
          [] a_SECC_processPLCLinkNotifications_002();
          [] pt_SLAC_Port.receive
          {
             setverdict(fail, "Invalid message was received within " &
                              "sleeping phase on port 'pt_SLAC_Port'.");
          }
          [] pt_V2G_UDP_SDP_Port.receive
          {
             setverdict(fail, "Invalid message was received within " &
                              "sleeping phase on port 'pt_V2G_UDP_SDP_Port'.");
          }
      }
      return getverdict;
  }
}
```

F.5.2 EVCC functions

```
module TestBehavior_EVCC_CommonBehavior {

    import from DataStructure_HAL_61851 all;
    import from Templates_CMN_HAL61851 all;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Templates_CMN_V2GTcpTlsPortControl all;
import from Templates_CMN_V2GMessage all;
import from ComponentsAndPorts all;
import from DataStructure_V2G_TCP_TLS_Port_Control all;
import from urn_iso_15118_2_2013_MsgBody all;
import from urn_iso_15118_2_2013_MsgDef all;
import from DataStructure_SDP all;
import from Services_EVCC_PLCLinkStatus all;
import from Templates_CMN_SessionStop all;

altstep a_EVCC_Unexpected_Message_Content(verdicttype v_verdict, charstring v_message)
                                          runs on EVCC_Tester {

  [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive
   {
      tc_V2G_SECC_Sequence_Timer.stop;
      setverdict(v_verdict, v_message);
   }
}

altstep a_EVCC_Unexpected_Message(verdicttype v_verdict, charstring v_message)
                                  runs on EVCC_Tester {

  [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive
   {
       setverdict(v_verdict, v_message);
   }
 }

altstep a_EVCC_Unexpected_V2G_Message(verdicttype v_verdict, charstring v_message,
                                      template BodyElement_group body)
                                      runs on EVCC_Tester {

  [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(?,
                                        md_CMN_BodyType_001(complement(body)))) {

      tc_V2G_SECC_Sequence_Timer.stop;
      setverdict(v_verdict, v_message);
   }
}

altstep a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(verdicttype v_verdict, charstring v_message,
                                                 template BodyElement_group body1,
                                                 template BodyElement_group body2)
                                                 runs on EVCC_Tester {

  [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(?,
                                        md_CMN_BodyType_001(complement(body1, body2)))) {

      tc_V2G_SECC_Sequence_Timer.stop;
      setverdict(v_verdict, v_message);
   }
}

altstep a_EVCC_Unexpected_V2G_Message_PIXIT_Var3(verdicttype v_verdict, charstring v_message,
                                                 template BodyElement_group body1,
                                                 template BodyElement_group body2,
                                                 template BodyElement_group body3)
                                                 runs on EVCC_Tester {

  [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(?,
                                        md_CMN_BodyType_001(complement(body1, body2, body3)))) {

      tc_V2G_SECC_Sequence_Timer.stop;
      setverdict(v_verdict, v_message);
   }
}

altstep a_EVCC_Timeout(verdicttype v_verdict) runs on EVCC_Tester {

   [] tc_V2G_SECC_Sequence_Timer.timeout
    {
       setverdict(v_verdict, "Nothing received. V2G_EVCC_Msg_Timer was triggered.");
    }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 altstep a_EVCC_TCPConnection_Status_Listener(verdicttype v_verdict,
                                              charstring v_message)
                                              runs on EVCC_Tester {

   [] pt_V2G_TCP_TLS_Port_Control.receive(md_CMN_CMN_tcpTlsStatusRes_001(
                                        e_disconnectPort, ACK_SUCCESS, true, *)) {
         setverdict(v_verdict,v_message);
         vc_isTcpPortClosed := true;
     }
 }

 altstep a_EVCC_No_SDP_Message(charstring v_message, verdicttype v_verdict)
                               runs on EVCC_Tester {

    [] pt_V2G_UDP_SDP_Port.receive
     {
        tc_V2G_SECC_CommunicationSetup_Timer.stop;
        setverdict(v_verdict, v_message);
     }
 }

altstep a_EVCC_SDP_Timeout(verdicttype v_verdict) runs on EVCC_Tester {

    [] tc_V2G_SECC_CommunicationSetup_Timer.timeout {

       setverdict(v_verdict, "V2G_EVCC_CommunicationSetup_Timer " &
                               "was triggered. No SDP request message was " &
                               "received within 'V2G_EVCC_CommunicationSetup_Timeout'.");
    }
}

 altstep a_SDP_Message() runs on EVCC_Tester {

     [] pt_V2G_UDP_SDP_Port.receive(SDP_Message:?) {

          repeat;
     }
 }

 altstep a_EVCC_TCP_Connection_Termination_Timeout() runs on EVCC_Tester {

   [] tc_CMN_TCP_Connection_Termination_Timer.timeout {
       log("TCP connection was >>NOT<< terminated by the SUT, " &
             "tc_CMN_TCP_Connection_Termination_Timer was triggered.");
   }
 }

 altstep a_EVCC_StateB_Shutdown_Timeout() runs on EVCC_Tester {

   [] tc_EVCC_StateB_Shutdown_Timer.timeout {
       log("CP State B was not detected. " &
           "EVCC_StateB_Shutdown_Timer was triggered.");
   }
 }

function isError() return boolean {

 if(getverdict == error) {
     return true;
 }
 return false;
}

function f_EVCC_startSleepingPhase(in float v_time) runs on EVCC_Tester
                                                        return verdicttype {

    timer t_sleep := v_time;
    deactivate(vc_Default_SDP_Message);

    t_sleep.start;
    alt {
        [] t_sleep.timeout {
           log("Sleeping phase was terminated by the Test System.")
        }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_EVCC_processPLCLinkNotifications_001();
            [] pt_SLAC_Port.receive
            {
               setverdict(fail, "Invalid message was received within " &
                                "sleeping phase on port 'pt_SLAC_Port'.");
            }
            [] pt_V2G_UDP_SDP_Port.receive
            {
               setverdict(fail, "Invalid message was received within " &
                                "sleeping phase on port 'pt_V2G_UDP_SDP_Port'.");
            }
         }
         return getverdict;
      }

      altstep a_EVCC_CMN_SessionStop_Termination() runs on EVCC_Tester {

           []  pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                    md_CMN_MessageHeaderType_001(
                                                    vc_SessionID),
                                                    md_CMN_BodyType_001(
                                                    md_CMN_CMN_SessionStopReq_001(
                                                    terminate)))) {

            setverdict(pass, "SessionStopReq message received. " &
                             "The EVCC will terminate the session.");

            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
                                              md_CMN_BodyType_001(
                                              md_CMN_CMN_SessionStopRes_001(oK))));
        }
      }
   }
```

F.6 Library functions

```
module LibFunctions_15118_2 {

    import from Pics_15118 all;
    import from Timer_15118 all;
    import from Pixit_15118 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from XSDAUX all;
    import from Templates_CMN_Security all;
    import from urn_iso_15118_2_2013_MsgHeader language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" {type V2G_Message;}
    import from urn_iso_15118_2_2013_MsgBody all;
    import from Templates_CMN_V2GMessage all;
    import from DataStructure_Security all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from Services_Security all;
    import from urn_iso_15118_2_2013_X509Def all;
    import from Pics_15118_2 all;
    import from TTlibrary_Logging all;
    import from ComponentsAndPorts all;
    import from DataStructure_PICS_15118 all;

    group PhysicalValues {

        function f_CMN_PhysicalValue_GetValue(in PhysicalValueType v) return float {
          var float ret;
          if (v.multiplier < 0) { ret :=(int2float(v.value_) *
                                        (1.0/(f_CMN_Pow(v.multiplier*-1, 10.0))));}
          else if (v.multiplier > 0) { ret := (int2float(v.value_) *
                                            (f_CMN_Pow(v.multiplier, 10.0)));}
          else { ret := int2float(v.value_); }

          return ret;
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
function f_CMN_Pow(in integer v_multiplier, in float v_basis) return float {
  var float ret := v_basis;
  for (var integer i:=0; i<v_multiplier-1; i:=i + 1) {
    ret := ret * v_basis;
  }
  return ret;
}

function f_CMN_AsbFlt(in float v_first, in float v_second) return float {
  if(v_first == v_second){
    return 0.0;
  }
  if(v_first > v_second){
    return v_first - v_second;
  }else{
    return v_second - v_first;
  }
}

function f_CMN_AsbInt(in integer v_first, in integer v_second) return integer {
  if(v_first == v_second){
    return 0;
  }
  if(v_first > v_second){
    return v_first - v_second;
  }else{
    return v_second - v_first;
  }
}

function f_CMN_Float2Int(in float v_float, in integer v_position) return integer {

  for (var integer i:=0; i<v_position; i:=i + 1) {
    v_float := v_float * 10.0;
  }
  return float2int(v_float);
}

function f_CMN_Sqrt(in float v_Float) return float {

    var float v_x := 1.0;
    var float v_xAlt := 0.0;
    var float v_epsilon := 0.00001;

    while (f_CMN_AsbFlt(v_x, v_xAlt) > v_epsilon) {
        v_xAlt := v_x;
        v_x := (v_x * v_x + v_Float) / (2.0 * v_x);
    }

    return v_x;
}

function f_check_PhysicalValue_valueRange(in charstring v_text,
                                          in PhysicalValueType v_physicalValue,
                                          in UnitSymbolType v_unit,
                                          in float v_maxValue,
                                          in verdicttype v_vct) {

  if(f_CMN_PhysicalValue_GetValue(v_physicalValue) > v_maxValue) {
        setverdict(v_vct, "The physical value '" & toString<charstring>(v_text) &
                          "' has an invalid maximum value (" &
                          toString<float>(f_CMN_PhysicalValue_GetValue(v_physicalValue)) &
                          toString<UnitSymbolType>(v_unit) &
                          " > " & toString<float>(v_maxValue) &
                          toString<UnitSymbolType>(v_unit) & ").");
  }
     if(v_physicalValue.unit != v_unit) {
        setverdict(v_vct, "The physical value '" & toString<charstring>(v_text) &
                          "' has an invalid unit (" &
                          toString<UnitSymbolType>(v_physicalValue.unit) &
                          " != " & toString<UnitSymbolType>(v_unit) & ").");
     }
}

function f_CMN_getMinimumVal(in PhysicalValueType v_val1,
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                in PhysicalValueType v_val2)
                                return PhysicalValueType {

      if(f_CMN_PhysicalValue_GetValue(v_val1) < f_CMN_PhysicalValue_GetValue(v_val2)) {
        return v_val1;
      } else {
        return v_val2;
      }
    }

    function f_checkNegativeSign(in integer v_value) return integer {

        if(v_value >= 0) {
            return v_value;
        } else {
            return 0;
        }
    }
}

group generalFunctions {

    type record ListOfFloat {
        record length (0 .. 10) of float v_float
    }

    type record ListOfBoolean {
        record length (0 .. 10) of boolean v_boolean
    }

    external function fx_validateEVSEID(in EvseIDType v_evseID,
                                        in charstring v_protocol) return boolean;

    external function fx_setPhysicalValueType(in float v_float,
                                              in charstring v_unit) return PhysicalValueType;

    external function fx_floor(in float v_float, in integer v_decimalPoints) return float;

    external function fx_ceil(in float v_float, in integer v_decimalPoints) return float;

    external function fx_getTimeAsLong() return integer;

    function sleep(float time) {
        timer t := time;
        t.start;
        t.timeout;
    }

    function f_rnd_SessionID(in integer p_lower, in integer p_upper) return octetstring {
        return int2oct(float2int(rnd() * int2float(p_upper - p_lower + 1)) + p_lower, 8)
    }

    function f_checkWakeUpConfig(in WakeUp v_wakeUp) return verdicttype {

        var verdicttype v_verdict := pass;

        select (v_wakeUp) {

         case (tsInitiated) {

             if(PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp) {
                 log("Valid wake-up configuration is used.");
             } else {
                 setverdict(inconc, "Inalid wake-up configuration is used. " &
                                    "PIXIT_CMN_CMN_WakeUp > PICS_CMN_CMN_WakeUp is NOT " &
                                    "fulfilled as defined in the test objective.");
                v_verdict := getverdict;
             }
         }
         case (sutInitiated) {

            if(PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp) {
                log("Valid wake-up configuration is used.");
            } else {
                setverdict(inconc, "Inalid wake-up configuration is used. " &
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    "PIXIT_CMN_CMN_WakeUp < PICS_CMN_CMN_WakeUp is NOT " &
                                    "fulfilled as defined in the test objective.");
                v_verdict := getverdict;
            }
        }
        case (sutInitiatedZeroPow) {

            if(par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp) {
               log("Valid wake-up configuration is used.");
            } else {
                setverdict(inconc, "Inalid wake-up configuration is used. " &
                                    "par_SECC_Pmax0W < PICS_CMN_CMN_WakeUp is NOT " &
                                    "fulfilled as defined in the test objective.");
                v_verdict := getverdict;
            }
        }
      }
      return v_verdict;
    }
}

group security {

function f_validateRootCertificateIDLength(in template (value) BodyType p_BodyType,
                                           in verdicttype v_vct) {

    var integer maxLengthOfRootCertificateIDString := 40;
    var integer v_counter := 0;
    var urn_iso_15118_2_2013_MsgDataTypes.ListOfRootCertificateIDsType v_rootCertificateID_list;

    if (ischosen(p_BodyType.bodyElement.certificateInstallationReq)) {
        log("CertificateInstallationReq WAS RECEIVED -> CHECK ListOfRootCertificateIDsType " &
            "-> X509IssuerSerialType -> x509IssuerName length <= 40");
        v_rootCertificateID_list := p_BodyType.bodyElement.
                                    certificateInstallationReq.listOfRootCertificateIDs;
    } else if (ischosen(p_BodyType.bodyElement.certificateUpdateReq))    {
        log("CertificateUpdateReq WAS RECEIVED -> CHECK ListOfRootCertificateIDsType " &
            "-> X509IssuerSerialType -> x509IssuerName length <= 40");
        v_rootCertificateID_list := p_BodyType.bodyElement.
                                    certificateUpdateReq.listOfRootCertificateIDs; //RECORD OF
    }

    var integer sizeOfRootCertificateID_list := lengthof(v_rootCertificateID_list.
                                                         rootCertificateID_list); //max 20 entries

    for (v_counter := 0; v_counter < sizeOfRootCertificateID_list; v_counter:= v_counter +1) {
          log("Counter: " & toString<integer>(v_counter));
          var X509IssuerSerialType v_X509IssuerSerialType := v_rootCertificateID_list.
                                                             rootCertificateID_list[v_counter];
          var charstring v_x509IssuerNameCharstring :=  v_X509IssuerSerialType.x509IssuerName;
          var integer v_sizeOfX509IssuerNameCharstring := lengthof(v_x509IssuerNameCharstring);
          log("X509IssuerName: " & toString<charstring>(v_x509IssuerNameCharstring));
          log("Size of X509IssuerName: " & toString<integer>(v_sizeOfX509IssuerNameCharstring));
          if (v_sizeOfX509IssuerNameCharstring > 40) {
                setverdict(v_vct, "RootCertificateID string size is larger " &
                                  "than 40 [V2G2-356].");
          }
    }
}

function f_generateDigestFromBodyType(in template (value) BodyType p_BodyType)
                                      return XSDAUX.base64Binary {
    return fx_convertOctetstring2Base64Binary(fx_messageDigest(
                                              "SHA-256", fx_canonicalExi(
                                              encvalue(p_BodyType))));
}

function f_calculateIdDigestMapFromBodyType(in template (value) BodyType p_BodyType)
                                            return idDigestValueMap {
    var idDigestValueMap resultIdDigestValueMap := {};

    if (ischosen(p_BodyType.bodyElement.authorizationReq)) {
        var charstring hashedExiHexstringAsBase64BinaryString :=
        fx_convertOctetstring2Base64Binary(fx_messageDigest("SHA-256",
                                           fx_canonicalExi(encvalue(p_BodyType))));
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        resultIdDigestValueMap := {
            [0] := {p_BodyType.bodyElement.authorizationReq.id,
                    hashedExiHexstringAsBase64BinaryString}
        };
    } else if (ischosen(p_BodyType.bodyElement.meteringReceiptReq)) {
        var charstring hashedExiHexstringAsBase64BinaryString :=
        fx_convertOctetstring2Base64Binary(fx_messageDigest("SHA-256",
                                        fx_canonicalExi(encvalue(p_BodyType))));
        resultIdDigestValueMap := {
            [0] := {p_BodyType.bodyElement.meteringReceiptReq.id,
                    hashedExiHexstringAsBase64BinaryString}
        };
    } else if (ischosen(p_BodyType.bodyElement.certificateInstallationReq)) {
        var charstring hashedExiHexstringAsBase64BinaryString :=
        fx_convertOctetstring2Base64Binary(fx_messageDigest("SHA-256",
                                        fx_canonicalExi(encvalue(p_BodyType))));
        resultIdDigestValueMap := {
            [0] := {p_BodyType.bodyElement.certificateInstallationReq.id,
                    hashedExiHexstringAsBase64BinaryString}
        };
    } else if (ischosen(p_BodyType.bodyElement.certificateInstallationRes)) {
        var CertificateInstallationRes v_certificateInstallationRes :=
        p_BodyType.bodyElement.certificateInstallationRes;
        var CertificateChainType v_contractSignatureCertChain :=
        v_certificateInstallationRes.contractSignatureCertChain;
        var ContractSignatureEncryptedPrivateKeyType
        v_contractSignatureEncryptedPrivateKey :=
        v_certificateInstallationRes.contractSignatureEncryptedPrivateKey;
        var DiffieHellmanPublickeyType v_dHpublickey :=
        v_certificateInstallationRes.dHpublickey;
        var EMAIDType v_eMAID := v_certificateInstallationRes.eMAID;

        var integer v_counterIdDigestValueMap :=0;
        var octetstring certificateChainTypeAsOctetstring :=
        fx_marshallTtcn3CertificateChainTypeWithJaxb(
        v_contractSignatureCertChain);
        var bitstring encodedContractSignatureCertChain :=
        oct2bit(certificateChainTypeAsOctetstring);

        resultIdDigestValueMap[v_counterIdDigestValueMap] := {
                            v_contractSignatureCertChain.id,
                            fx_convertOctetstring2Base64Binary(
                            fx_messageDigest("SHA-256",
                            fx_canonicalExi(
                            encodedContractSignatureCertChain))) };

        v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;

        var octetstring contractSignatureEncryptedPrivateKeyTypeAsOctetstring :=
        fx_marshallTtcn3ContractSignatureEncryptedPrivateKeyTypeWithJaxb(
        v_contractSignatureEncryptedPrivateKey);
        var bitstring encodedContractSignatureEncryptedPrivateKeyTypeAsBitstring :=
        oct2bit(contractSignatureEncryptedPrivateKeyTypeAsOctetstring);

        resultIdDigestValueMap[v_counterIdDigestValueMap] := {
                v_contractSignatureEncryptedPrivateKey.id,
                fx_convertOctetstring2Base64Binary(fx_messageDigest(
                "SHA-256", fx_canonicalExi(
                encodedContractSignatureEncryptedPrivateKeyTypeAsBitstring)))};

        v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;

        var octetstring diffieHellmanPublickeyTypeAsOctetstring :=
        fx_marshallTtcn3DiffieHellmanPublickeyTypeWithJaxb(v_dHpublickey);
        var bitstring encodedDiffieHellmanPublickeyTypeAsBitstring :=
        oct2bit(diffieHellmanPublickeyTypeAsOctetstring);

        resultIdDigestValueMap[v_counterIdDigestValueMap] := {
                            v_dHpublickey.id,
                            fx_convertOctetstring2Base64Binary(
                            fx_messageDigest("SHA-256",
                            fx_canonicalExi(
                            encodedDiffieHellmanPublickeyTypeAsBitstring))) };

        v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        var octetstring eMAIDTypeAsOctetstring :=
        fx_marshallTtcn3EMAIDTypeWithJaxb(v_eMAID);
        var bitstring eMAIDTypeAsBitstring := oct2bit(eMAIDTypeAsOctetstring);

        resultIdDigestValueMap[v_counterIdDigestValueMap] := {
                            v_eMAID.id,
                            fx_convertOctetstring2Base64Binary(fx_messageDigest(
                            "SHA-256", fx_canonicalExi(eMAIDTypeAsBitstring))) };

    } else if (ischosen(p_BodyType.bodyElement.certificateUpdateReq)) {
        var charstring hashedExiHexstringAsBase64BinaryString :=
        fx_convertOctetstring2Base64Binary(fx_messageDigest("SHA-256",
                                        fx_canonicalExi(
                                        encvalue(p_BodyType))));
        resultIdDigestValueMap := {
            [0] := {p_BodyType.bodyElement.certificateUpdateReq.id,
                    hashedExiHexstringAsBase64BinaryString}
        };
    } else if (ischosen(p_BodyType.bodyElement.certificateUpdateRes)) {
        var CertificateUpdateRes v_certificateUpdateRes :=
        p_BodyType.bodyElement.certificateUpdateRes;
        var CertificateChainType v_contractSignatureCertChain :=
        v_certificateUpdateRes.contractSignatureCertChain;
        var ContractSignatureEncryptedPrivateKeyType
        v_contractSignatureEncryptedPrivateKey :=
        v_certificateUpdateRes.contractSignatureEncryptedPrivateKey;
        var DiffieHellmanPublickeyType v_dHpublickey := v_certificateUpdateRes.
                                                        dHpublickey;
        var EMAIDType v_eMAID := v_certificateUpdateRes.eMAID;
        var integer v_counterIdDigestValueMap :=0;

        var octetstring certificateChainTypeAsOctetstring :=
        fx_marshallTtcn3CertificateChainTypeWithJaxb(v_contractSignatureCertChain);
        var bitstring encodedContractSignatureCertChain :=
        oct2bit(certificateChainTypeAsOctetstring);

        resultIdDigestValueMap[v_counterIdDigestValueMap] := {
                            v_contractSignatureCertChain.id,
                            fx_convertOctetstring2Base64Binary(
                            fx_messageDigest("SHA-256",
                            fx_canonicalExi(encodedContractSignatureCertChain))) };

        v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;

        var octetstring contractSignatureEncryptedPrivateKeyTypeAsOctetstring :=
        fx_marshallTtcn3ContractSignatureEncryptedPrivateKeyTypeWithJaxb(
        v_contractSignatureEncryptedPrivateKey);
        var bitstring encodedContractSignatureEncryptedPrivateKeyTypeAsBitstring :=
        oct2bit(contractSignatureEncryptedPrivateKeyTypeAsOctetstring);

        resultIdDigestValueMap[v_counterIdDigestValueMap] := {
                v_contractSignatureEncryptedPrivateKey.id,
                fx_convertOctetstring2Base64Binary(
                fx_messageDigest("SHA-256",
                fx_canonicalExi(
                encodedContractSignatureEncryptedPrivateKeyTypeAsBitstring))) };

        v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;

        var octetstring diffieHellmanPublickeyTypeAsOctetstring :=
        fx_marshallTtcn3DiffieHellmanPublickeyTypeWithJaxb(v_dHpublickey);
        var bitstring encodedDiffieHellmanPublickeyTypeAsBitstring :=
        oct2bit(diffieHellmanPublickeyTypeAsOctetstring);

        resultIdDigestValueMap[v_counterIdDigestValueMap] := {
                            v_dHpublickey.id,
                            fx_convertOctetstring2Base64Binary(
                            fx_messageDigest("SHA-256",
                            fx_canonicalExi(
                            encodedDiffieHellmanPublickeyTypeAsBitstring))) };

        v_counterIdDigestValueMap := v_counterIdDigestValueMap+1;

        var octetstring eMAIDTypeAsOctetstring :=
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        fx_marshallTtcn3EMAIDTypeWithJaxb(v_eMAID);
        var bitstring eMAIDTypeAsBitstring := oct2bit(eMAIDTypeAsOctetstring);

        resultIdDigestValueMap[v_counterIdDigestValueMap] := {
                            v_eMAID.id,
                            fx_convertOctetstring2Base64Binary(
                            fx_messageDigest("SHA-256",
                            fx_canonicalExi(eMAIDTypeAsBitstring))) };
    }else if (ischosen(p_BodyType.bodyElement.chargeParameterDiscoveryRes)) {
        //MISSING SalesTariffs in ChargeParameterDisoveryRes
        //log("Found BodyType >> ChargeParameterDiscoveryRes => get SalesTariffs");
        var ChargeParameterDiscoveryResType v_chargeParameterDiscoverRes :=
        p_BodyType.bodyElement.chargeParameterDiscoveryRes;

        if (ispresent(v_chargeParameterDiscoverRes.sASchedules)) {
            var integer v_counter;
            var SASchedules_group v_sASchedules :=
            v_chargeParameterDiscoverRes.sASchedules;
            var SAScheduleListType.sAScheduleTuple_list v_sAScheduleTuple_list :=
            v_sASchedules.sAScheduleList.sAScheduleTuple_list; //RECORD OF
            var integer sizeOfScheduleTuple_list := lengthof(v_sAScheduleTuple_list);

            //get salesTariff for each element in list
            for (v_counter := 0; v_counter < sizeOfScheduleTuple_list; v_counter:= v_counter +1) {
                //log("Counter: ", v_counter);
                var SalesTariffType v_salesTariff := v_sAScheduleTuple_list[v_counter].
                                                    salesTariff;
                //log("==> v_salesTariff: ", v_salesTariff);
                var charstring v_salesTariffId := v_salesTariff.id;
                //log("==> v_salesTariffId: ", v_salesTariffId);
                var octetstring salesTariffAsOctetstring :=
                fx_marshallTtcn3V2GSalesTariffTypeWithJaxb(v_salesTariff);
                var bitstring encodedSalesTariffBitString :=
                oct2bit(salesTariffAsOctetstring);
                var charstring digestValue:= fx_convertOctetstring2Base64Binary(
                                                fx_messageDigest("SHA-256",
                                                fx_canonicalExi(
                                                encodedSalesTariffBitString)));
                //log("==> calculated digestValue: ", digestValue);

                //NOW store id and digest value
                resultIdDigestValueMap[v_counter] := { v_salesTariffId,    digestValue};
            }
        }
    }else {
        log("Given BodyType not relevant for Signature verification");
    }
    return resultIdDigestValueMap;
}

function f_verifySignature(in template (value) BodyType p_bodyType,
                           in template (value) SignatureType p_signatureType,
                           in template (value) X509CertificateType p_certificate,
                           in verdicttype v_vct)
                           return boolean {

    var integer v_counter_for_referenceList;
    var integer v_counter_for_idDigestValueMap;
    var charstring currentIdInReferenceList;
    var charstring currentIdInIdDigestValueMap;
    var charstring currentDigestValueInReferenceList;
    var charstring currentDigestValueInIdDigestValueMap;

    //    contains calculated digestValues for all ids in bodytype message
    var idDigestValueMap calcIdDigestValueMap :=
        f_calculateIdDigestMapFromBodyType(p_bodyType);
    //    contains digestValues and signatureValue from header
    var SignedInfo signedInfo := p_signatureType.signedInfo;
    var integer v_sizeOfCalcIdDigestValueMap := lengthof(calcIdDigestValueMap);

    //BUGFIX: IF no id digest elements were FOUND, return false
    if (v_sizeOfCalcIdDigestValueMap == 0) {
        setverdict(v_vct, "No digest values were found in BodyType: ", p_bodyType);
        return false;
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
//Digest comparison procedure:
//    a) Iterate over elements in calculated id-digest-hashmap        (-> outer loop)
//        b) Iterate over elements in signedInfo                         (-> inner loop)
for (v_counter_for_idDigestValueMap := 0;
     v_counter_for_idDigestValueMap < lengthof(calcIdDigestValueMap);
     v_counter_for_idDigestValueMap := v_counter_for_idDigestValueMap +1) {

    var boolean matchedDigestValue := false;
    var boolean matchedId := false;

    currentIdInIdDigestValueMap :=
    calcIdDigestValueMap[v_counter_for_idDigestValueMap].id;
    currentDigestValueInIdDigestValueMap :=
    calcIdDigestValueMap[v_counter_for_idDigestValueMap].digestValue;

     for (v_counter_for_referenceList := 0;
           v_counter_for_referenceList < lengthof(signedInfo.reference_list);
           v_counter_for_referenceList := v_counter_for_referenceList +1) {

        var ReferenceType refType :=
            signedInfo.reference_list[v_counter_for_referenceList];
        var charstring refTypeUri := refType.uRI;
        var charstring refTypeUriWithoutHashtag :=
        substr(refTypeUri, 1, (lengthof(refTypeUri)-1)); //remove #

        currentIdInReferenceList := refTypeUriWithoutHashtag;
        currentDigestValueInReferenceList := refType.digestValue;

        // 2-stepped comparison
        //    -> first     check if IDs do exist
        //  -> second     compare digestValue for matched ID
        if (currentIdInReferenceList == currentIdInIdDigestValueMap) {
            matchedId := true;

            if (currentDigestValueInReferenceList ==
                currentDigestValueInIdDigestValueMap) {
                //log("DigestValue MATCH: ", currentDigestValueInReferenceList);
                matchedDigestValue := true;
            } else {
                log("Calculated digestValue ",currentDigestValueInIdDigestValueMap,
                    " does NOT match digestValue in SignedInfo ",
                    currentDigestValueInReferenceList, " for Id ",
                    currentIdInReferenceList);
            }
        }
    }

    //NOTICE: Already return on first "ID mismatch"
    if (not matchedId) {
        setverdict(v_vct, "ID ", currentIdInIdDigestValueMap,
                   " could NOT be found in SignedInfo element ",
                   signedInfo);
        return false;
    }

    //NOTICE: Already return on first digestValue mismatch!!!
    if (not matchedDigestValue) {
        setverdict(v_vct, "Digest MISMATCH");
        return false;
    }
};


var charstring signatureCharstringValue := p_signatureType.
                                           signatureValue.base;
var bitstring encodedSignedInfoType :=encvalue(signedInfo);

var octetstring exiEncodedSignedInfoTypeOctetstring :=
fx_canonicalExi(encodedSignedInfoType);

var boolean verified :=
fx_verifySignedInfoDigest("SHA256withECDSA",
                          exiEncodedSignedInfoTypeOctetstring,
                          signatureCharstringValue,
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                p_certificate.raw_Certificate
                                );

    if(verified) {
        setverdict(pass, "Signature verification was successful.");
    }

    return verified;
}

function f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
         in template (value) BodyType p_BodyType,
         in template (value) idDigestValueMap salesTariffIdDigestValueMap)
         return LocalReferenceList {

    var integer v_counter;
    var LocalReferenceList list;

    for (v_counter := 0; v_counter < lengthof(
         salesTariffIdDigestValueMap); v_counter:= v_counter +1) {

        list[v_counter] := md_ReferenceType_001(
                           salesTariffIdDigestValueMap[v_counter].id,
                           salesTariffIdDigestValueMap[v_counter].digestValue);
    }
    return list;
}

function f_getPrivateKey(in template (value) idPrivateKeyMap v_IdPrivateKeyMap,
                         in template (value) charstring v_id)
                         return charstring {

    var charstring v_privateKey;
    var integer v_counter;

    for (v_counter := 0; v_counter < lengthof(
         v_IdPrivateKeyMap); v_counter:= v_counter +1) {

         if(v_IdPrivateKeyMap[v_counter].id == v_id) {
            v_privateKey := v_IdPrivateKeyMap[v_counter].privateKey;
         }
    }
    if(not(ispresent(v_privateKey))) {
        setverdict(inconc, "The selected private key could not be found.");
    }
    return v_privateKey;
}

function f_getCertificate(in template (value) idCertificateMap v_IdCertificateMap,
                         in template (value) charstring v_id)
                         return X509CertificateType {

    var X509CertificateType v_certificate;
    var integer v_counter;

    for (v_counter := 0; v_counter < lengthof(
         v_IdCertificateMap); v_counter:= v_counter +1) {

         if(v_IdCertificateMap[v_counter].id == v_id) {
            v_certificate := v_IdCertificateMap[v_counter].certificate;
         }
    }
    if(not(ispresent(v_certificate))) {
        setverdict(inconc, "The selected certificate could not be found.");
    }
    return v_certificate;
}

function f_checkSignatureConstFields(Signature v_signature,
                                     charstring cc_algorithm_method_1,
                                     charstring cc_algorithm_method_2,
                                     charstring cc_algorithm_method_3,
                                     in verdicttype v_vct){

    var charstring v_algorithm := v_signature.signedInfo.canonicalizationMethod.algorithm;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(v_algorithm != cc_algorithm_method_1) {
                setverdict(v_vct, "Invalid algorithm method was detected.");
        }

        v_algorithm := v_signature.signedInfo.signatureMethod.algorithm;
        if(v_algorithm != cc_algorithm_method_2) {
                setverdict(v_vct, "Invalid algorithm method was detected.");
        }

        for (var integer i:=0; i<sizeof(v_signature.signedInfo.reference_list); i:=i + 1) {
            for (var integer j:=0; j<sizeof(v_signature.signedInfo.reference_list[i].
                                        transforms.transform_list); j:=j + 1) {
                v_algorithm := v_signature.signedInfo.reference_list[i].
                               transforms.transform_list[j].algorithm;
                if(v_algorithm != cc_algorithm_method_1) {
                        setverdict(v_vct, "Invalid algorithm method was detected.");
                }
            }
            v_algorithm := v_signature.signedInfo.reference_list[i].
                           digestMethod.algorithm;
            if(v_algorithm != cc_algorithm_method_3) {
                    setverdict(v_vct, "Invalid algorithm method was detected.");

            }
        }
    }
    }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex G
(normative)

Function specifications for 15118-2 V2GTP

G.1 General information

This annex covers all *test behaviors* for V2GTP in the ATS.

G.2 SECC functions

```
module TestBehavior_SECC_V2GTPSessionSetup {

    import from Timer_15118_2 all;
    import from ComponentsAndPorts all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_CMN_SupportedAppProtocol all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_SessionSetup all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_CMN_V2GTPMessage all;
    import from DataStructure_V2GTP all;

    function f_SECC_CMN_TB_VTB_V2GTPSessionSetup_001() runs on SECC_Tester return verdicttype {

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2GTP_Message_001(
                                md_CMN_V2GTP_header_TYPE_001('01'H,'FE'H,'8001'H, omit),
                                md_CMN_V2GTP_payload_TYPE_002(
                                md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_SessionSetupReq_001(vc_eVCCID))))));

        alt {
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2GTP_Message_001(
                                        md_CMN_V2GTP_header_TYPE_001('01'H,'FE'H,'8001'H,?),
                                        md_CMN_V2GTP_payload_TYPE_002(
                                        md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(?),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_SessionSetupRes_001(?,?,*))))){

                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"V2GTP (SessionSetupRes) message was correct.");
            }
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                          "terminated by the SUT.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received anything, but not " &
                                                       "the expected V2GTP message.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_CommunicationSetup_Timeout(fail);
        }
        return getverdict;
    }

    function f_SECC_CMN_TB_VTB_V2GTPSessionSetup_002(in template V2GTP_header_TYPE v_v2gtpHeader)
                                                 runs on SECC_Tester return verdicttype {

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2GTP_Message_001(v_v2gtpHeader,
                                md_CMN_V2GTP_payload_TYPE_002(
                                md_CMN_V2G_Message_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_SessionSetupReq_001(vc_eVCCID))))));

        alt {
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2GTP_Message_001(
                                                md_CMN_V2GTP_header_TYPE_001(
                                                '01'H,'FE'H,'8001'H,?),
                                                md_CMN_V2GTP_payload_TYPE_002(
                                                md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(?),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_SessionSetupRes_001(?,?,*)))))) {

                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(fail, "V2GTP (SessionSetupRes) message was not expected.");
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                        "terminated by the SUT.") {
                setverdict(pass, "SessionSetupReq message with invalid header " &
                                 "information was detected.");
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received anything, but not the " &
                                                       "expected V2GTP message.");
            [] a_SECC_Timeout(pass) {
                setverdict(pass, "SessionSetupReq message with invalid header " &
                                 "information was detected.");
            }
            [] a_SECC_CommunicationSetup_Timeout(fail);
        }
        return getverdict;
    }
}



module TestBehavior_SECC_V2GTPSDP {

   import from ComponentsAndPorts all;
   import from Timer_15118_2 all;
   import from Templates_CMN_SDP all;
   import from Templates_CMN_V2GTPMessage all;
   import from DataStructure_SDP all;
   import from DataStructure_V2GTP all;
   import from TestBehavior_SECC_CommonBehavior all;
   import from Services_HAL_61851_1 all;

   function f_SECC_CMN_TB_VTB_V2GTPSDP_001() runs on SECC_Tester return verdicttype {

       var V2GTP_Message v_responseMessage;
       var integer v_count := 0;

       while(v_count < vc_maxRepetitionSDP){

           tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
           pt_V2G_UDP_SDP_Port.send(md_CMN_V2GTP_Message_001(
                                    md_CMN_V2GTP_header_TYPE_001('01'H,'FE'H,'9000'H, omit),
                                    md_CMN_V2GTP_payload_TYPE_001(
                                    md_CMN_SDP_Request_TYPE_001('10'H,'00'H))));
           v_count := v_count + 1;

           alt {
               [] pt_V2G_UDP_SDP_Port.receive(md_CMN_V2GTP_Message_001(
                                                md_CMN_V2GTP_header_TYPE_001(
                                                '01'H,'FE'H,'9001'H,'00000014'H),
                                                md_CMN_V2GTP_payload_TYPE_001(
                                                md_CMN_SDP_Response_TYPE_001(?, ?, ?, ?))))
                                                -> value v_responseMessage {

                   tc_EVCC_SDP_Timer.stop;
                      vc_V2G_Port_IpAddress := v_responseMessage.v2gtp_payload.content.sdp_message.
                                                content.sdp_Response.secc_IPaddress;
                      vc_V2G_Port_PortNumber := v_responseMessage.v2gtp_payload.content.sdp_message.
                                                  content.sdp_Response.secc_Port;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                v_count := vc_maxRepetitionSDP;
                setverdict(pass,"V2GTP Header message was correct. " &
                                "(SDP Response message).");
            }
            [] pt_V2G_UDP_SDP_Port.receive {
                v_count := vc_maxRepetitionSDP;
                setverdict(fail, "Invalid V2GTP Header (SDP Response message) was received.");
            }
            [] tc_EVCC_SDP_Timer.timeout {
               // send a new SDP request message if v_count < vc_maxRepetitionSDP
               if(v_count == vc_maxRepetitionSDP){
                   setverdict(fail,"SDP repetition maximum is reached. SDP process " &
                                   "shall be stopped.");
               }
            }
            [] a_SECC_CommunicationSetup_Timeout(fail);
        }
     }
     return getverdict;
  }

  function f_SECC_CMN_TB_VTB_V2GTPSDP_002(in template V2GTP_header_TYPE v_v2gtpHeader)
                                          runs on SECC_Tester return verdicttype {

      var V2GTP_Message v_responseMessage;

      tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
      pt_V2G_UDP_SDP_Port.send(md_CMN_V2GTP_Message_001(v_v2gtpHeader,
                               md_CMN_V2GTP_payload_TYPE_001(
                               md_CMN_SDP_Request_TYPE_001('10'H,'00'H))));

      alt {
          [] pt_V2G_UDP_SDP_Port.receive(md_CMN_V2GTP_Message_001(
                                         md_CMN_V2GTP_header_TYPE_001(
                                         '01'H,'FE'H,'9001'H,'00000014'H),
                                         md_CMN_V2GTP_payload_TYPE_001(
                                         md_CMN_SDP_Response_TYPE_001(?, ?, ?, ?))))
                                         -> value v_responseMessage {

              tc_EVCC_SDP_Timer.stop;
              setverdict(fail,"Invalid V2GTP header was not ignored.");
          }
          [] pt_V2G_UDP_SDP_Port.receive {
              setverdict(fail,"Invalid V2GTP header was not ignored.");
          }
          [] tc_EVCC_SDP_Timer.timeout {
              setverdict(pass,"Invalid V2GTP header was ignored.");
          }
          [] a_SECC_CommunicationSetup_Timeout(fail);
      }
      return getverdict;
   }
}
```

G.3 EVCC functions

```
module TestBehavior_EVCC_V2GTPSessionSetup {

    import from ComponentsAndPorts all;
    import from Timer_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from Templates_CMN_SupportedAppProtocol all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from Templates_CMN_SessionSetup all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_CMN_V2GTPMessage all;
    import from urn_iso_15118_2_2010_AppProtocol language "XSD" all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from DataStructure_V2GTP all;
import from Timer_15118 all;
import from LibFunctions_15118_2 all;

function f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_001() runs on EVCC_Tester return verdicttype {

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2GTP_Message_001(
                                md_CMN_V2GTP_header_TYPE_001('01'H, 'FE'H, '8001'H, omit),
                                md_CMN_V2GTP_payload_TYPE_004(
                                md_CMN_CMN_SupportedAppProtocolRes_001(
                                oK_SuccessfulNegotiation,(vc_SchemaID)))));
    alt
    {
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2GTP_Message_001(
                                          md_CMN_V2GTP_header_TYPE_001('01'H, 'FE'H,
                                                                       '8001'H, ?),
                                          md_CMN_V2GTP_payload_TYPE_002(
                                          md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(?),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_SessionSetupReq_001(?))))))
        {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(pass, "V2GTP (SessionSetupReq) message was correct.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received anything, but not the " &
                                                   "expected V2GTP message.");
        [] a_EVCC_Timeout(fail);
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_V2GTPSessionSetup_002(in template V2GTP_header_TYPE v_v2gtpHeader)
                                                 runs on EVCC_Tester return verdicttype {

   vc_SessionID := f_rnd_SessionID(1,429496);
   tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
   var boolean v_tcpTerminated := false;

   // send Session Setup response message with invalid header information
   pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2GTP_Message_001(
                                   v_v2gtpHeader, md_CMN_V2GTP_payload_TYPE_002(
                                   md_CMN_V2G_Message_001(md_CMN_MessageHeaderType_001(
                                   vc_SessionID), md_CMN_BodyType_001(
                                   md_CMN_CMN_SessionSetupRes_001(
                                   oK_NewSessionEstablished, vc_EvseId,
                                   fx_getTimeAsLong()))))));
   timer t_wait;

   // Wait until SUT message timeout
   t_wait.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq -
                  par_CMN_Transmission_Delay);
   alt {
       [] t_wait.timeout;
       [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                     "terminated by the SUT " &
                                                     "before timeout has expired.") {
           v_tcpTerminated := true;
           setverdict(pass, "SessionSetupRes message with invalid header " &
                            "information was detected.");
       }
       [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2GTP_Message_001(
                                         md_CMN_V2GTP_header_TYPE_001(?, ?, ?, ?),
                                         md_CMN_V2GTP_payload_TYPE_002(
                                         md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(?),
                                         md_CMN_BodyType_001(
                                         md_CMN_CMN_ServiceDiscoveryReq_003(*,*))))) {
           tc_V2G_SECC_Sequence_Timer.stop;
           setverdict(fail, "V2GTP (ServiceDiscoveryReq) message was not expected.");
       }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
                setverdict(fail, "No V2G message was expected.");
            }
        }

        if(getverdict == pass and not v_tcpTerminated) {

            tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                            "terminated by the SUT.") {
                   setverdict(pass, "SessionSetupRes message with invalid header " &
                                     "information was detected.");
                }
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2GTP_Message_001(
                                                        md_CMN_V2GTP_header_TYPE_001(?, ?, ?, ?),
                                                        md_CMN_V2GTP_payload_TYPE_002(
                                                        md_CMN_V2G_Message_001(
                                                        md_CMN_MessageHeaderType_001(?),
                                                        md_CMN_BodyType_001(
                                                        md_CMN_CMN_ServiceDiscoveryReq_003(*,*)))))) {
                     tc_V2G_SECC_Sequence_Timer.stop;
                     setverdict(fail, "V2GTP (ServiceDiscoveryReq) message was not expected.");
                }
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
                     setverdict(fail, "No V2G message was expected.");
                }
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                     setverdict(fail,"SUT has not terminated the TCP connection " &
                                     "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
                [] a_EVCC_Timeout(fail);
            }
        }
        return getverdict;
    }
}



module TestBehavior_EVCC_V2GTPSDP {

    import from ComponentsAndPorts all;
    import from Timer_15118_2 all;
    import from Templates_CMN_SDP all;
    import from Templates_CMN_V2GTPMessage all;
    import from DataStructure_SDP all;
    import from DataStructure_V2GTP all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851_1 all;
    import from Configurations_15118_2 all;

    function f_EVCC_CMN_TB_VTB_V2GTPSDP_001() runs on EVCC_Tester return verdicttype {

        alt
        {
            [] pt_V2G_UDP_SDP_Port.receive(md_CMN_V2GTP_Message_001(
                                           md_CMN_V2GTP_header_TYPE_001('01'H, 'FE'H,
                                                                        '9000'H, '00000002'H),
                                           md_CMN_V2GTP_payload_TYPE_001(
                                           md_CMN_SDP_Request_TYPE_001(?, ?))))
            {
                setverdict(pass, "V2GTP Header message was correct. (SDP Request Message).");
            }
            [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail);
            [] a_EVCC_SDP_Timeout(fail);
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_V2GTPSDP_002(in template V2GTP_header_TYPE v_v2gtpHeader)
                                            runs on EVCC_Tester return verdicttype {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        var V2GTP_Message v_requestMessage;
        var boolean v_repetition := true;

        while(v_repetition){
            alt
            {
                [] pt_V2G_UDP_SDP_Port.receive(md_CMN_V2GTP_Message_001(
                                                md_CMN_V2GTP_header_TYPE_001('01'H, 'FE'H,
                                                                                '9000'H, '00000002'H),
                                                md_CMN_V2GTP_payload_TYPE_001(
                                                md_CMN_SDP_Request_TYPE_001(?, ?))))
                                                -> value v_requestMessage {

                   vc_TransportProtocol := v_requestMessage.v2gtp_payload.content.
                                                sdp_message.content.sdp_Request.transportProtocol;
                   vc_Security := v_requestMessage.v2gtp_payload.content.sdp_message.
                                    content.sdp_Request.security;

                   if(not v_repetition){
                       setverdict(pass, "SDP response message with invalid header " &
                                           "information was ignored. " &
                                           "EVCC has repeated SDP request message.");
                       break;
                   }

                   // send SDP response message with invalid header information
                   pt_V2G_UDP_SDP_Port.send(md_CMN_V2GTP_Message_001(
                                                v_v2gtpHeader, md_CMN_V2GTP_payload_TYPE_001(
                                                md_CMN_SDP_Response_TYPE_001(
                                                par_SECC_IPaddress, vc_V2G_Port_PortNumber,
                                                vc_Security, vc_TransportProtocol))));
                   v_repetition := false;
                   repeat;

                }
                [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail){
                   v_repetition := false;
                }
                [] a_EVCC_SDP_Timeout(fail){
                   v_repetition := false;
                }
            }
        }
        return getverdict;
    }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex H
(normative)

Function specifications for 15118-2 SDP messages

H.1 General information

This annex covers all *test behaviors* for SDP in the ATS.

H.2 SECC functions

```
module TestBehavior_SECC_SDP {

    import from ComponentsAndPorts all;
    import from Timer_15118_2 all;
    import from Templates_CMN_SDP all;
    import from DataStructure_SDP all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Services_HAL_61851_1 all;

    function f_SECC_CMN_TB_VTB_SDP_001(in Security_TYPE v_security,
                                       in verdicttype v_vct)
                                       runs on SECC_Tester
                                       return verdicttype {

        var integer v_count := 0;
        var SDP_Message v_responseMessage;

        while(v_count < vc_maxRepetitionSDP){
            tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
            pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Request_TYPE_001(v_security,'00'H)) to cc_multicast;
            v_count := v_count + 1;

            alt {
                [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(
                                                ?, (49152 .. 65535), v_security, '00'H))
                                                -> value v_responseMessage {
                   tc_EVCC_SDP_Timer.stop;
                   vc_V2G_Port_IpAddress := v_responseMessage.content.sdp_Response.secc_IPaddress;
                   vc_V2G_Port_PortNumber := v_responseMessage.content.sdp_Response.secc_Port;
                   vc_Security := v_responseMessage.content.sdp_Response.security;
                   v_count := vc_maxRepetitionSDP;
                   setverdict(pass,"SDP Response message was correct.");
                }

                [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(
                                                ?, (49152 .. 65535), ?, '00'H))
                                                -> value v_responseMessage {
                   tc_EVCC_SDP_Timer.stop;
                   v_count := vc_maxRepetitionSDP;
                   setverdict(inconc,"SUT did not support the requested transmission security.");
                }
                [] pt_V2G_UDP_SDP_Port.receive {
                   v_count := vc_maxRepetitionSDP;
                   setverdict(v_vct, "Invalid SDP response message was received.");
                }
                [] tc_EVCC_SDP_Timer.timeout {
                   // send a new SDP request message if v_count < vc_maxRepetitionSDP
                   if(v_count == vc_maxRepetitionSDP){
                       setverdict(v_vct,"SDP repetition maximum is reached. SDP process " &
                                        "shall be stopped.");
                   }
                }
                [] a_SECC_CommunicationSetup_Timeout(v_vct);
            }
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        return getverdict;
    }

    function f_SECC_CMN_TB_VTB_SDP_002(in Security_TYPE v_security) runs on SECC_Tester
                                                                        return verdicttype {

        alt {
            [] tc_V2G_SECC_CommunicationSetup_Timer.timeout;
            [] pt_V2G_UDP_SDP_Port.receive {
                setverdict(fail, "No SDP message was expected.");
            }
        }

        tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
        pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Request_TYPE_001(v_security,'00'H))
                                                              to cc_multicast;

        alt {
            [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(
                                        ?, (49152 .. 65535), ?, ?)){
               tc_EVCC_SDP_Timer.stop;
               setverdict(fail,"The SUT did not stop the SDP server.");
            }
            [] tc_EVCC_SDP_Timer.timeout {

               setverdict(pass,"No SDP response was received. The SUT has " &
                                "stopped the SDP server.");
            }
        }
        return getverdict;
    }

    function f_SECC_CMN_TB_VTB_SDP_003() runs on SECC_Tester return verdicttype {

        var integer v_count := 0;
        var SDP_Message v_responseMessage;

        while(v_count < vc_maxRepetitionSDP){
            tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
            pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Request_TYPE_001('00'H,'00'H)) to cc_multicast;
            v_count := v_count + 1;

            alt {
                [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(
                                        ?, (49152 .. 65535), '10'H, '00'H))
                                        -> value v_responseMessage {
                   tc_EVCC_SDP_Timer.stop;
                   vc_V2G_Port_IpAddress := v_responseMessage.content.sdp_Response.secc_IPaddress;
                   vc_V2G_Port_PortNumber := v_responseMessage.content.sdp_Response.secc_Port;
                   vc_Security := v_responseMessage.content.sdp_Response.security;
                   v_count := vc_maxRepetitionSDP;
                   setverdict(pass,"SDP Response message was correct.");
                }
                [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(
                                        ?, (49152 .. 65535), ?, '00'H))
                                        -> value v_responseMessage {
                   tc_EVCC_SDP_Timer.stop;
                   v_count := vc_maxRepetitionSDP;
                   setverdict(inconc,"SUT supports the requested transmission security " &
                                     "'secured with TLS'. Check the PIXIT configuration.");
                }
                [] pt_V2G_UDP_SDP_Port.receive {
                  v_count := vc_maxRepetitionSDP;
                  setverdict(fail, "Invalid SDP response message was received.");
                }
                [] tc_EVCC_SDP_Timer.timeout {
                   // send a new SDP request message if v_count < vc_maxRepetitionSDP
                   if(v_count == vc_maxRepetitionSDP){
                       setverdict(fail,"SDP repetition maximum is reached. SDP process " &
                                       "shall be stopped.");
                   }
                }
                [] a_SECC_CommunicationSetup_Timeout(fail);
            }
        }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        return getverdict;
    }
}
```

H.3 EVCC functions

```
module TestBehavior_EVCC_SDP {

    import from LibFunctions_15118_2 all;
    import from ComponentsAndPorts all;
    import from Templates_CMN_SDP all;
    import from DataStructure_SDP all;
    import from Timer_15118_2 all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Templates_EVCC_SupportedAppProtocol all;
    import from Configurations_15118_2 all;
    import from Services_Security all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Templates_CMN_Security all;
    import from urn_iso_15118_2_2013_X509Def all;
    import from Pics_15118_2 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from DataStructure_V2G_TCP_TLS_Port_Control all;
    import from TTlibrary_Logging all;

    function f_EVCC_CMN_TB_VTB_SDP_001(in template Security_TYPE v_security,
                                       in verdicttype v_vct)
                                       runs on EVCC_Tester return verdicttype {

        var SDP_Message v_requestMessage;

        alt {

            []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(v_security, '00'H))
                                       -> value v_requestMessage
                                          sender vc_sut_ipAddress.ipAddress {

             vc_TransportProtocol := v_requestMessage.content.sdp_Request.transportProtocol;
             vc_Security := v_requestMessage.content.sdp_Request.security;
             tc_V2G_SECC_CommunicationSetup_Timer.stop;
             setverdict(pass,"SDP Request message was correct.");
            }
            []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?, '00'H)) {

             tc_V2G_SECC_CommunicationSetup_Timer.stop;
             setverdict(inconc,"SUT did not request the correct transmission security.");
            }
            [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", v_vct);
            [] a_EVCC_SDP_Timeout(v_vct);
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_SDP_002() runs on EVCC_Tester return verdicttype {

        var SDP_Message v_requestMessage;
        var integer v_count := 0;

        alt {
            []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?, '00'H))
                                       -> value v_requestMessage {

             v_count := v_count + 1;
             log("Number of received SDP messages: ", v_count);
             repeat;
            }
            [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail);
            [v_count == vc_maxRepetitionSDP] tc_V2G_SECC_CommunicationSetup_Timer.timeout {
               setverdict(pass,"The SUT has exactly sent 'par_maxRepetitionSDP' SECC Discovery " &
                               "Request messages within 'V2G_SECC_CommunicationSetup_Timer'.");
            }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
          [v_count < vc_maxRepetitionSDP]
           tc_V2G_SECC_CommunicationSetup_Timer.timeout {
             setverdict(fail,"The SUT has sent less than 'par_maxRepetitionSDP' SECC Discovery " &
                             "Request messages within 'V2G_SECC_CommunicationSetup_Timer'.");
          }
          [v_count > vc_maxRepetitionSDP]
           tc_V2G_SECC_CommunicationSetup_Timer.timeout {
             setverdict(fail,"The SUT has sent more than 'par_maxRepetitionSDP' SECC Discovery " &
                             "Request messages within 'V2G_SECC_CommunicationSetup_Timer'.");
          }
      }
      return getverdict;
}

function f_EVCC_CMN_TB_VTB_SDP_003() runs on EVCC_Tester return verdicttype  {

    var SDP_Message v_requestMessage;
    var integer v_count := 0;
    var SDP_Message v_requestMessageSDP;
    var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_responseMessage;
    var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;

    alt {
        []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?,'00'H))
                                      -> value v_requestMessage
                                       sender vc_sut_ipAddress.ipAddress {

          tc_EVCC_SDP_Timer.start(par_EVCC_SDP_Timeout);
          v_count := v_count + 1;
          log("Number of received SDP messages: " & toString<integer>(v_count));

          vc_TransportProtocol := v_requestMessage.content.sdp_Request.transportProtocol;
          vc_Security := v_requestMessage.content.sdp_Request.security;
          if(vc_Security == cc_hexTcp) {
             vc_v2g_tcp_tls_parameter := md_CMN_CMN_TcpParameter_001(vc_sut_ipAddress.ipAddress,
                                                                     vc_V2G_Port_PortNumber,
             vc_Security);
          } else if(vc_Security == cc_hexTls) {

             vc_v2g_tcp_tls_parameter := md_CMN_CMN_TlsParameter_001(vc_sut_ipAddress.ipAddress,
                                                                     vc_V2G_Port_PortNumber,
                                                                     vc_Security, vc_cipherSuites,
                                                                     vc_enabledProtocol,
                                                                     md_CertificateChainType_002(
                                                                     vc_leafCert, vc_subCert2,
                                                                     vc_subCert1));
          } else {log("Invalid security type was chosen.")};

             if(v_count == 1) {
                f_EVCC_CMN_PR_InitConfiguration_002(system, fail);
             }

          tc_EVCC_SDP_Timer.timeout;

          if(v_count > 1) {

              tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

              pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Response_TYPE_001(
                                       par_SECC_IPaddress, vc_V2G_Port_PortNumber,
                                       vc_Security, vc_TransportProtocol));

              pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Response_TYPE_001(
                                    par_SECC_IPaddress, vc_V2G_Port_PortNumber,
                                    vc_Security, vc_TransportProtocol));

              alt {
                  //SDP Handling if more than one message is received from SUT
                  []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?, '00'H))
                                                -> value v_requestMessageSDP {
                       v_count := v_count + 1;
                          log("Number of received SDP messages: " & toString<integer>(v_count));
                       pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Response_TYPE_001(
                                                par_SECC_IPaddress,
                                                vc_V2G_Port_PortNumber,
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                vc_Security, vc_TransportProtocol));
                         repeat;
                    }
                    [] pt_V2G_TCP_TLS_Port_Control.receive(md_CMN_CMN_tcpTlsStatusRes_001(
                                                    e_connectPort,?,?,*))
                                                    -> value v_responseMessage {

                        var boolean isClosed := v_responseMessage.result.portStatus.closed;
                        ackType := v_responseMessage.ack;
                        if(ackType == ACK_SUCCESS) {
                           setverdict(pass, "TCP or TLS handshake was successful.");
                           repeat;
                        } else if(ackType == ACK_FAILED) {
                           setverdict(fail, "TCP or TLS handshake was not successful.");
                        }
                    }
                    []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                      mw_EVCC_CMN_SupportedAppProtocolReq_001()) {

                        tc_V2G_SECC_Sequence_Timer.stop;
                        setverdict(pass,"SupportedAppProtocolReq message was correct. " &
                                           "The SDP client is able to handle " &
                                        " asynchronously arriving SECC Discovery " &
                                        "Response messages");
                    }
                    [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                                "but cannot be decoded.");
                    [] a_EVCC_Unexpected_Message_Content(fail, "Received " &
                                                                "SupportedAppProtocolReq " &
                                                                "message, but with invalid " &
                                                                "data content.");
                    [] a_EVCC_Timeout(fail);
                 }
              }
              else {
                 repeat;
              }
           }
           [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail);
           [] a_EVCC_SDP_Timeout(fail);
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_SDP_004() runs on EVCC_Tester return verdicttype  {

        var SDP_Message v_requestMessage;

        alt {
            []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?,'00'H))
                                          -> value v_requestMessage {

             vc_TransportProtocol := v_requestMessage.content.sdp_Request.transportProtocol;
             vc_Security := v_requestMessage.content.sdp_Request.security;

             pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Request_TYPE_001(vc_Security, vc_TransportProtocol));

             alt {
                []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?,'00'H)) {
                  setverdict(pass,"The SDP client did not reply to SDP Request message.");
                }
                [] pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Response_TYPE_001(?, (49152 .. 65535),
                                                                    vc_Security, '00'H)) {
                  setverdict(fail,"The SDP client did reply to SDP Request message.");
                }
                [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail);
                [] a_EVCC_SDP_Timeout(fail);
                }
            }
            [] a_EVCC_No_SDP_Message("Received anything, but no SDP_Message", fail);
            [] a_EVCC_SDP_Timeout(fail);
        }
        return getverdict;
     }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex I
(normative)

Function specifications for 15118-2 V2G application layer messages

I.1 General information

This annex covers all *test behaviors* for the V2G application layer messages in the ATS.

I.2 SECC functions

I.2.1 V2G protocol handshake

I.2.1.1 SECC functions for SupportedAppProtocol

```
module TestBehavior_SECC_SupportedAppProtocol {

    import from Services_HAL_61851_1 all;
    import from urn_iso_15118_2_2010_AppProtocol all;
    import from Configurations_15118_2 all;
    import from ComponentsAndPorts all;
    import from Timer_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_CMN_SupportedAppProtocol all;
    import from Templates_SECC_SupportedAppProtocol all;
    import from Templates_CMN_SessionSetup all;
    import from Templates_CMN_V2GMessage all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Services_HAL_61851_1 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Pixit_15118_2 all;
    import from Timer_15118 all;
    import from LibFunctions_15118_2 all;

    function f_SECC_CMN_TB_VTB_SupportedAppProtocol_001(in verdicttype v_vct)
                                                        runs on SECC_Tester
                                                        return verdicttype {

        var SupportedAppProtocolRes v_responseMessage;
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SupportedAppProtocolReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_SupportedAppProtocolReq_001(
                                    md_CMN_CMN_AppProtocolType_001(
                                    vc_Supported_ISO_Namespace,
                                    vc_versionNumberMajor,
                                    vc_versionNumberMinor,10,1)));

        alt {

            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_CMN_SupportedAppProtocolRes_001(
                                                m_SECC_CMN_validResponseCodes(),*))
                                                -> value v_responseMessage {
                tc_V2G_EVCC_Msg_Timer.stop;
                if(v_responseMessage.responseCode == oK_SuccessfulNegotiation){
                    setverdict(pass,"SupportedAppProtocolRes message was correct.");
                }
                else {
                   setverdict(pass,"SupportedAppProtocolRes message was correct. " &
                                  "Minor version deviation was detected.");
                }
            }
            [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                        "terminated by the SUT.");
            [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
```

 © ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                                                 "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(v_vct, "Received SupportedAppProtocolRes message, " &
                                                 "but with invalid data content.");
        [] a_SECC_Timeout(v_vct);
        [] a_SECC_CommunicationSetup_Timeout(v_vct);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SupportedAppProtocol_002() runs on SECC_Tester
                                                       return verdicttype {

    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SupportedAppProtocolReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_SupportedAppProtocolReq_001(
                                 md_CMN_CMN_AppProtocolType_001(
                                 "urn:iso:15118:2:2014:MsgDef",3,
                                 vc_versionNumberMinor,1,1)));

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_CMN_SupportedAppProtocolRes_002(
                                                failed_NoNegotiation)) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"SupportedAppProtocolRes message with responseCode " &
                            "'fAILED_NoNegotiation' was received.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                         e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received SupportedAppRes message, " &
                                                    "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_CommunicationSetup_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a SupportedAppProtocolRes " &
                        "message with responseCode 'fAILED_NoNegotiation' " &
                        "or other message data content was not conform. ";
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SupportedAppProtocol_003(in timer t_wait) runs on SECC_Tester
                                                                     return verdicttype {

    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var float v_timeShift := 1.0;

    // Wait until sequence timeout in SUT should expire
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                        "terminated by the SUT " &
                                        "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        sleep(v_timeShift - par_CMN_Transmission_Delay);


        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SupportedAppProtocol_004() runs on SECC_Tester return verdicttype {

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SupportedAppProtocolReq);

    vc_versionNumberMinor := PIXIT_SECC_CMN_VersionNumberMinor + 1;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(vc_versionNumberMinor < vc_versionNumberMajor) {

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_CMN_SupportedAppProtocolReq_001(
                                        md_CMN_CMN_AppProtocolType_001(
                                        vc_Supported_ISO_Namespace,
                                        vc_versionNumberMajor,
                                        vc_versionNumberMinor,10,1)));

            alt {

                []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_CMN_SupportedAppProtocolRes_001(
                                            oK_SuccessfulNegotiationWithMinorDeviation,*)) {
                     tc_V2G_EVCC_Msg_Timer.stop;
                     setverdict(pass,"SupportedAppProtocolRes message was correct. " &
                                    "Minor version deviation was detected.");
                }
                [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                             "was misleadingly " &
                                                             "terminated by the SUT.");
                [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                             "but cannot be decoded.");
                [] a_SECC_Unexpected_Message_Content(fail, "Received SupportedAppProtocolRes " &
                                                            "message, but with invalid data " &
                                                            "content.");
                [] a_SECC_Timeout(fail);
                [] a_SECC_CommunicationSetup_Timeout(fail);
            }
        } else {
            setverdict(inconc, "The selected VersionNumberMinor is not smaller " &
                               "than VersionNumberMajor.");
        }
        return getverdict;
    }
}
```

I.2.2 V2G messages

I.2.2.1 SECC functions for SessionSetup

```
module TestBehavior_SECC_SessionSetup {

    import from Timer_15118_2 all;
    import from ComponentsAndPorts {type SECC_Tester;};
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_SessionSetup all;
    import from Templates_CMN_V2GMessage all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Services_HAL_61851_1 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118 all;

    function f_is_SessionID_not_zero(SessionIDType sessionID) return boolean {

        if(oct2int(sessionID)==0){
            return false;
        }
        else {
            return true;
        }
    }

    function f_SECC_CMN_TB_VTB_SessionSetup_001(in verdicttype v_vct)
                                            runs on SECC_Tester
                                            return verdicttype {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var V2G_Message v_responseMessage;
    var charstring v_message;

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(md_CMN_CMN_SessionSetupReq_001(
                                vc_eVCCID))));

    alt {
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(?),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_SessionSetupRes_001(
                                        oK_NewSessionEstablished,?,*))))
                                        -> value v_responseMessage {

            tc_V2G_EVCC_Msg_Timer.stop;
            vc_SessionID := v_responseMessage.header.sessionID;
            vc_EvseId := v_responseMessage.body.bodyElement.sessionSetupRes.eVSEID;
            log("EVSEID: " & toString<charstring>(vc_EvseId));
            if(vc_EvseId != cc_EvseId_NotSup) {
                var boolean result := fx_validateEVSEID(vc_EvseId, "ISO");
                if(not result) {
                        setverdict(v_vct,"Invalid EVSEID syntax was detected.");
                }
            }
            if(getverdict == pass) {
                if(not f_is_SessionID_not_zero(vc_SessionID)){
                        setverdict(v_vct,"Invalid SessionID was detected.");
                }
                else{
                    setverdict(pass,"SessionSetup message was correct.");
                }
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {sessionSetupRes := ?});
        [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(v_vct, "Received SessionSetupRes message, " &
                                                    "but with invalid data content.");
        [] a_SECC_Timeout(v_vct);
        [] a_SECC_CommunicationSetup_Timeout(v_vct);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SessionSetup_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryReq_002(omit))));
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            []    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(?),
                                                md_CMN_BodyType_001(
                                                mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                                fAILED_SequenceError)))) {

                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                                "'fAILED_SequenceError' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {serviceDiscoveryRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received SessionSetupRes message, " &
                                                        "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_CommunicationSetup_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                            "message with responseCode 'fAILED_SequenceError' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    function f_SECC_CMN_TB_VTB_SessionSetup_003() runs on SECC_Tester return verdicttype {

        timer t_wait;
        var boolean isShutdownOSC := false;
        var boolean isClosedTCP := false;
        var float v_timeShift := 1.0;

        // Wait until sequence timeout in SUT should expire
        t_wait.start(par_V2G_SECC_Sequence_Timeout -
                     v_timeShift);
        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));

        alt {
            [] t_wait.timeout;
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                            "terminated by the SUT " &
                                            "before timeout has expired.");
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
                setverdict(fail, "No V2G message was expected.");
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
    }

    if(getverdict == pass) {

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SessionSetup_004() runs on SECC_Tester return verdicttype {

    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    // Wait until V2G_SECC_CommunicationSetup timeout in SUT should expire
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

    alt {
        [] tc_V2G_SECC_CommunicationSetup_Timer.timeout;
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            tc_V2G_SECC_CommunicationSetup_Timer.stop;
            setverdict(fail, "No V2G message was expected.");
        }
    }

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                isClosedTCP := true;
                tc_CMN_TCP_Connection_Termination_Timer.stop;
                if(isShutdownOSC == false) {
                   logWarn("SECC oscillator was not shut down before the " &
                           "TCP connection was terminated.");}
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
        return getverdict;
    }


    function f_SECC_CMN_TB_VTB_SessionSetup_005(in verdicttype v_vct)
                                                runs on SECC_Tester
                                                return verdicttype {

        var V2G_Message v_responseMessage;
        var charstring v_message;

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_SessionSetupReq_001(vc_eVCCID))));

        alt {
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_SessionSetupRes_001(
                                                oK_OldSessionJoined,?,*))))
                                                -> value v_responseMessage {

                tc_V2G_EVCC_Msg_Timer.stop;
                vc_EvseId := v_responseMessage.body.bodyElement.sessionSetupRes.eVSEID;
                log("EVSEID: " & toString<charstring>(vc_EvseId));
                if(vc_EvseId != cc_EvseId_NotSup) {
                var boolean result := fx_validateEVSEID(vc_EvseId, "ISO");
                    if(not result) {
                            setverdict(v_vct,"Invalid EVSEID syntax was detected.");
                    }
                }
                if(getverdict == pass) {
                    setverdict(pass,"SessionSetupRes message was correct. A ResponseCode " &
                                    "with 'oK_OldSessionJoined' was received " &
                                    "after paused session.");
                }
            }

            [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                           "terminated by the SUT.");
            [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {sessionSetupRes := ?});
            [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(v_vct, "Received SessionSetupRes message, " &
                                                        "but with invalid data content.");
            [] a_SECC_Timeout(v_vct);
            [] a_SECC_CommunicationSetup_Timeout(v_vct);
        }
        return getverdict;
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    function f_SECC_CMN_TB_VTB_SessionSetup_006(in boolean v_isZero) runs on SECC_Tester
                                                                     return verdicttype {

        var V2G_Message v_responseMessage;
        var charstring v_message;
        var SessionIDType v_SessionID_old;

        v_SessionID_old := vc_SessionID;
        if(v_isZero) {
            vc_SessionID := '0000000000000000'O;
        }
        else {
            vc_SessionID := f_rnd_SessionID(1,429496);
        }

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_SessionSetupReq_001(vc_eVCCID))));

        alt {
            []    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(?),
                                              md_CMN_BodyType_001(
                                              md_CMN_CMN_SessionSetupRes_001(
                                              oK_NewSessionEstablished,?,*))))
                                              -> value v_responseMessage {

                tc_V2G_EVCC_Msg_Timer.stop;
                vc_SessionID := v_responseMessage.header.sessionID;
                vc_EvseId := v_responseMessage.body.bodyElement.sessionSetupRes.eVSEID;
                log("EVSEID: " & toString<charstring>(vc_EvseId));
                if(vc_EvseId != cc_EvseId_NotSup) {
                var boolean result := fx_validateEVSEID(vc_EvseId, "ISO");
                    if(not result) {
                        setverdict(fail,"Invalid EVSEID syntax was detected.");
                    }
                }
                if(getverdict == pass) {
                    if(f_is_SessionID_not_zero(vc_SessionID) and
                       (vc_SessionID != v_SessionID_old)){
                           setverdict(pass,"SessionSetupRes message was correct. A ResponseCode " &
                                     "with 'oK_NewSessionEstablished' was received " &
                                     "after paused session.");
                    } else {
                        setverdict(fail,"The received SessionID is not unequal " &
                                        "to '0' and not unequal to the SessionID value "
                                        "stored from the preceding V2G Communication "
                                        "Session.");
                    }
                }
            }
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                          "terminated by the SUT.");
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {sessionSetupRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received SessionSetupRes message, " &
                                                       "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_CommunicationSetup_Timeout(fail);
        }
        return getverdict;
    }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

I.2.2.2 SECC functions for ServiceDiscovery

```
module TestBehavior_SECC_ServiceDiscovery {

    import from Services_HAL_61851_1 all;
    import from Timer_15118_2 all;
    import from Pics_15118 all;
    import from Pics_15118_2 all;
    import from ComponentsAndPorts all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_SessionSetup all;
    import from Templates_CMN_Authorization all;
    import from Templates_CMN_SupportedAppProtocol all;
    import from Templates_CMN_V2GMessage all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from TTlibrary_Logging all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Timer_15118 all;
    import from LibFunctions_15118_2 all;

    function f_SECC_CMN_TB_VTB_ServiceDiscovery_001(in verdicttype v_vct)
                                                    runs on SECC_Tester
                                                    return verdicttype {

        var V2G_Message v_responseMessage;
        var charstring v_message;

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

        alt {
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_SECC_CMN_ServiceDiscoveryRes_002(oK, ?,?,*))))
                                          -> value v_responseMessage{
                tc_V2G_EVCC_Msg_Timer.stop;
                vc_supportedEnergyTransferMode := v_responseMessage.body.bodyElement.
                                                  serviceDiscoveryRes.chargeService.
                                                  supportedEnergyTransferMode;

                vc_paymentOptionList := v_responseMessage.body.bodyElement.
                                        serviceDiscoveryRes.paymentOptionList;

                var boolean v_isSelected := false;
                for (var integer i:=0; i<sizeof(vc_paymentOptionList.paymentOption_list); i:=i + 1) {
                    if(PICS_CMN_CMN_IdentificationMode == pnC) {
                        if(vc_paymentOptionList.paymentOption_list[i] == contract) {
                           vc_SelectedPaymentOption := vc_paymentOptionList.paymentOption_list[i];
                           v_isSelected := true;
                        }
                    }
                    else {
                        if(vc_paymentOptionList.paymentOption_list[i] == externalPayment) {
                           vc_SelectedPaymentOption := vc_paymentOptionList.paymentOption_list[i];
                           v_isSelected := true;
                        }
                    }
                }
                if(not v_isSelected){
                    setverdict(v_vct,"Invalid payment option was offered by the SUT.");
                }
                else{
                    vc_serviceList := v_responseMessage.body.bodyElement.
                                      serviceDiscoveryRes.serviceList;
                    vc_chargeService := v_responseMessage.body.bodyElement.
                                        serviceDiscoveryRes.chargeService;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                setverdict(pass,"ServiceDiscoveryRes message was correct.");
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                        "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {serviceDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(v_vct, "Received ServiceDiscoveryRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(v_vct);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDiscovery_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(m_CMN_CMN_AuthorizationReq_001)));

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_AuthorizationRes_001(
                                          fAILED_SequenceError, ?)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"AuthorizationRes with responseCode " &
                            "'fAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {authorizationRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received ServiceDiscoveryRes message, " &
                                                   "but with invalid data content.");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a AuthorizationRes " &
                        "message with responseCode 'fAILED_SequenceError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDiscovery_003() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);
    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(
                                  f_rnd_SessionID(1,429496)),
                                  md_CMN_BodyType_001(
                                  m_CMN_CMN_ServiceDiscoveryReq_001)));

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(?),
                                        md_CMN_BodyType_001(
                                        md_SECC_CMN_ServiceDiscoveryRes_002(
                                        fAILED_UnknownSession, ?,?,omit)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                            "'fAILED_UnknownSession' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                         e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                         {serviceDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                              "but cannot be decoded.");
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] a_SECC_Unexpected_Message_Content(fail, "Received ServiceDiscoveryRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                        "message with responseCode 'fAILED_UnknownSession' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDiscovery_004() runs on SECC_Tester return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var float v_timeShift := 1.0;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
                 v_timeShift);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                        "terminated by the SUT " &
                                        "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    setverdict(fail,"SUT has not terminated the TCP connection " &
                                    "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_SECC_CMN_TB_VTB_ServiceDiscovery_005(in verdicttype v_vct)
                                                    runs on SECC_Tester
                                                    return verdicttype {

        var V2G_Message v_responseMessage;
        var charstring v_message;

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

        alt {
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_SECC_CMN_ServiceDiscoveryRes_002(
                                            oK, ?,vc_chargeService,*))))
                                            -> value v_responseMessage{
                tc_V2G_EVCC_Msg_Timer.stop;
                vc_supportedEnergyTransferMode := v_responseMessage.body.bodyElement.
                                                  serviceDiscoveryRes.chargeService.
                                                  supportedEnergyTransferMode;

                vc_SelectedPaymentOption := v_responseMessage.body.bodyElement.serviceDiscoveryRes.
                                            paymentOptionList.paymentOption_list[0];
                if(PICS_CMN_CMN_IdentificationMode == pnC) {
                    if(vc_SelectedPaymentOption != contract) {
                       setverdict(v_vct, "Invalid payment option was chosen by the SUT.");
                    }
                }
                else {
                    if(vc_SelectedPaymentOption != externalPayment) {
                        setverdict(v_vct, "Invalid payment option was chosen by the SUT.");
                    }
                }
                vc_serviceList := v_responseMessage.body.bodyElement.serviceDiscoveryRes.
                                  serviceList;
                    setverdict(pass,"ServiceDiscoveryRes message was correct. The same " &
                                    "PaymentOption and ChargeService was selected " &
                                    "after paused session.");
            }
            [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                           "terminated by the SUT.");
            [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {serviceDiscoveryRes := ?});
            [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(v_vct, "Received ServiceDiscoveryRes message, " &
                                                        "but with invalid data content.");
            [] a_SECC_Timeout(v_vct);
        }
        return getverdict;
    }


}
```

I.2.2.3 SECC functions for ServiceDetail

```
module TestBehavior_SECC_ServiceDetail {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Pics_15118_2 all;
import from Timer_15118_2 all;
import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
import from ComponentsAndPorts all;
import from TestBehavior_SECC_CommonBehavior all;
import from Templates_SECC_ServiceDetail all;
import from Templates_CMN_ServiceDiscovery all;
import from Templates_SECC_ServiceDiscovery all;
import from Templates_CMN_V2GMessage all;
import from urn_iso_15118_2_2013_MsgDef all;
import from TTlibrary_Logging all;
import from Pics_15118 all;
import from Services_HAL_61851_1 all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from TTlibrary_Logging all;
import from Pixit_15118_2 all;
import from Timer_15118 all;
import from LibFunctions_15118_2 all;

function f_SECC_CMN_TB_VTB_ServiceDetail_001(in verdicttype v_vct)
                                            runs on SECC_Tester
                                            return verdicttype {

    var integer counter := 0;
    var charstring v_message;
    var V2G_Message v_responseMessage;
    var boolean v_result := true;

    if(ispresent(vc_serviceList)) {
        if(PIXIT_SECC_CMN_InternetAccess == internetAccess) {
            v_result := f_checkServiceCategory(internet, vc_serviceList);
        }
        if(v_result) {
            while (counter < sizeof(vc_serviceList.service_list)) {
                 var ServiceType v_service := vc_serviceList.service_list[counter].
                                                 serviceType;

                 f_checkServiceType(v_service, v_vct);

                 if(getverdict == pass) {
                     tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDetailReq);

                     pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(
                                                  vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  md_SECC_CMN_ServiceDetailReq_001(
                                                  v_service.serviceID))));
                     alt {
                        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                     md_CMN_MessageHeaderType_001(
                                                     vc_SessionID),
                                                     md_CMN_BodyType_001(
                                                     mdw_SECC_CMN_ServiceDetailRes_001(
                                                     v_service.serviceID))))
                                                     -> value v_responseMessage {

                             tc_V2G_EVCC_Msg_Timer.stop;
                             var ServiceParameterListType v_serviceParameterList :=
                             v_responseMessage.body.bodyElement.serviceDetailRes.
                             serviceParameterList;

                             f_checkServiceParameterList(v_serviceParameterList,
                                                          v_service, v_vct);

                             if(getverdict == pass) {
                                 setverdict(pass,"ServiceDetailRes message was correct.");
                             }
                             counter := counter +1;
                        }
                        [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                                       "misleadingly " &
                                                                       "terminated by " &
                                                                       "the SUT.") {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                counter := sizeof(vc_serviceList.service_list) + 1;
                                break;
                            }
                            [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message " &
                                                                    "was received.",
                                                            {serviceDetailRes := ?}) {

                                counter := sizeof(vc_serviceList.service_list) + 1;
                                break;
                            }
                            [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message " &
                                                                         "was received, " &
                                                                         "but cannot be decoded.") {
                                counter := sizeof(vc_serviceList.service_list) + 1;
                                break;
                            }
                            [] a_SECC_Unexpected_Message_Content(v_vct, "Received ServiceDetailRes " &
                                                                        "message, but with invalid " &
                                                                        "data content.") {
                                counter := sizeof(vc_serviceList.service_list) + 1;
                                break;
                            }
                            [] a_SECC_Timeout(v_vct) {
                                counter := sizeof(vc_serviceList.service_list) + 1;
                                break;
                            }
                        }
                    }
                    else {
                       counter := sizeof(vc_serviceList.service_list) + 1;
                    }
                }
            }
        }
        else {setverdict(inconc,"A serviceList is not available.");}
        return getverdict;
    }

function f_SECC_CMN_TB_VTB_ServiceDetail_002() runs on SECC_Tester return verdicttype {

    var integer counter := 0;
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(ispresent(vc_serviceList)) {
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryReq_002(omit))));
        alt {
             []
             a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
             }
             []    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                                fAILED_SequenceError)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                setverdict(pass,"ServiceDetailRes with responseCode " &
                                "'fAILED_SequenceError' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                     "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {serviceDiscoveryRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received ServiceDetailRes message, " &
                                                    "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                            "message with responseCode 'fAILED_SequenceError' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    } else {setverdict(inconc,"A serviceList is not available.");}
    return getverdict;
  }

function f_SECC_CMN_TB_VTB_ServiceDetail_003() runs on SECC_Tester return verdicttype {

    var integer counter := 0;
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(ispresent(vc_serviceList)) {
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDetailReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    f_rnd_SessionID(1,429496)),
                                    md_CMN_BodyType_001(
                                    md_SECC_CMN_ServiceDetailReq_001(
                                    vc_serviceList.service_list[0].serviceType.serviceID))));
        alt {
             []
             a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
             }
             []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                                                    md_CMN_MessageHeaderType_001(?),
                                                    md_CMN_BodyType_001(
                                                    mdw_SECC_CMN_ServiceDetailRes_003(
                                                    fAILED_UnknownSession)))) {

                  tc_V2G_EVCC_Msg_Timer.stop;
                  setverdict(pass,"ServiceDetailRes with responseCode " &
                                  "'fAILED_UnknownSession' is successful.");

                  pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                                e_getPortStatus));
                  isErrorMessage := true;
                  repeat;
               }
               [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                        "terminated by the SUT.") {
                   isClosedTCP := true;
                   tc_CMN_TCP_Connection_Termination_Timer.stop;
                   if(isShutdownOSC == false) {
                      logWarn("SECC oscillator was not shut down before the " &
                              "TCP connection was terminated.");}
                   pt_HAL_61851_Internal_Port.clear;
               }
               [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                                {serviceDetailRes := ?});
               [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
               [] a_SECC_Unexpected_Message_Content(fail, "Received ServiceDetailRes message, " &
                                                        "but with invalid data content.");
               [] a_SECC_Timeout(fail);
               [] a_SECC_TCP_Connection_Termination_Timeout() {
                  v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                  "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               }
            }
            if(not isErrorMessage) {
                v_charstring := v_charstring & "SUT has not sent a ServiceDetailRes " &
                                "message with responseCode 'fAILED_UnknownSession' " &
                                 "or other message data content was not conform. ";
            }
            if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
               setverdict(fail,v_charstring);
            }
        } else {setverdict(inconc,"A serviceList is not available.");}
        return getverdict;
    }

    function f_SECC_CMN_TB_VTB_ServiceDetail_004() runs on SECC_Tester return verdicttype {

        var integer counter := 0;
        var charstring v_message;
        var ServiceIDType v_invalidServiceID := 65535;
        var boolean isInvalid;
        var boolean isShutdownOSC := false;
        var boolean isClosedTCP := false;
        var boolean isErrorMessage := false;
        var charstring v_charstring := "";

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        if(ispresent(vc_serviceList)) {
            isInvalid := f_checkInvalidServiceID(v_invalidServiceID, vc_serviceList);
            if(isInvalid) {

                tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDetailReq);
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);

                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_BodyType_001(
                                        md_SECC_CMN_ServiceDetailReq_001(
                                        v_invalidServiceID))));
            alt {
                 []
                 a_Shutdown_Oscillator(pass){
                      isShutdownOSC := true;
                      if(isClosedTCP == false) {repeat;}
                 }
                 []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                mdw_SECC_CMN_ServiceDetailRes_003(
                                                fAILED_ServiceIDInvalid)))) {

                    tc_V2G_EVCC_Msg_Timer.stop;
                    setverdict(pass,"ServiceDetailRes with responseCode " &
                                   "'fAILED_ServiceIDInvalid' is successful.");

                    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                             e_getPortStatus));
                    isErrorMessage := true;
                    repeat;
                 }
                 [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT.") {
                     isClosedTCP := true;
                     tc_CMN_TCP_Connection_Termination_Timer.stop;
                     if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                                "TCP connection was terminated.");}
                     pt_HAL_61851_Internal_Port.clear;
                 }
                 [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                             {serviceDetailRes := ?});
                 [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                       "but cannot be decoded.");
                 [] a_SECC_Unexpected_Message_Content(fail, "Received ServiceDetailRes " &
                                                       "message, but with invalid " &
                                                       "data content.");
                 [] a_SECC_Timeout(fail);
                 [] a_SECC_TCP_Connection_Termination_Timeout() {
                    v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
                 }
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a ServiceDetailRes " &
                            "message with responseCode 'fAILED_ServiceIDInvalid' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    } else {setverdict(inconc,"A serviceList is not available.");}
    return getverdict;
}

function f_SECC_CMN_TB_VTB_ServiceDetail_005() runs on SECC_Tester return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var float v_timeShift := 1.0;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
                 v_timeShift);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                                        "terminated by the SUT " &
                                        "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_checkInvalidServiceID(ServiceIDType v_invalidServiceID,
                                 ServiceListType vc_serviceList) return boolean {

    for (var integer i:=0; i<sizeof(vc_serviceList.service_list); i:=i + 1) {
      if(v_invalidServiceID == vc_serviceList.service_list[0].serviceType.serviceID){
          setverdict(inconc,"Selected serviceID is not invalid.");
          return false;
      }
    }
    return true;
}

function f_checkServiceType(ServiceType v_service, in verdicttype v_vct) {

    if(v_service.serviceID == 1) {
            setverdict(v_vct, "Service Id '1' is reserved for the service " &
                              "category 'EVcharging'.");
    }
    else if(v_service.serviceID == 2)    {
        if(v_service.serviceCategory != contractCertificate) {
                setverdict(v_vct, "Invalid service category was used.");
        }
    }
    else if(v_service.serviceID == 3)    {
        if(v_service.serviceCategory != internet) {
                setverdict(v_vct, "Invalid service category was used.");
        }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  }
  else if(v_service.serviceID == 4)    {
      if(v_service.serviceCategory != otherCustom) {
              setverdict(v_vct, "Invalid service category was used.");
      }
  }
  else {
      log("Unknown service type was used.");
      log("Service ID: " & toString<integer>(v_service.serviceID));
      log("Service Name: " & toString<charstring>(v_service.serviceName));
      log("Service Category: " &
      toString<ServiceCategoryType>(v_service.serviceCategory));
  }
}

function f_checkServiceParameterList(ServiceParameterListType v_serviceParameterList,
                                     ServiceType v_service, in verdicttype v_vct) {

  if(v_service.serviceID == 2) {
      for (var integer i:=0; i<sizeof(v_serviceParameterList); i:=i + 1) {
          if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 1) {
              if(v_serviceParameterList.parameterSet_list[i].
              parameter_list[0].name != "Service") {
                      setverdict(v_vct, "Invalid ParameterName (Service) was used for the " &
                                        "parameter serviceID equals to 2 and " &
                                        "'parameterSetID' equals to 1.");
                      break;
              }
              else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
              choice.stringValue != "Installation") {
                      setverdict(v_vct, "Invalid stringValue was used for the parameter " &
                                          "serviceID equals to 2 and " &
                                          "'parameterSetID' equals to 1.");
                      break;
              }
          }
          else if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 2) {
              if(v_serviceParameterList.parameterSet_list[i].
              parameter_list[0].name != "Service") {
                      setverdict(v_vct, "Invalid ParameterName (Service) was used for the" &
                                        " parameter serviceID equals to 2 and " &
                                        "'parameterSetID' equals to 2.");
                      break;
              }
              else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
              choice.stringValue != "Update") {
                      setverdict(v_vct, "Invalid stringValue was used for the parameter " &
                                          "serviceID equals to 2 and " &
                                         "'parameterSetID' equals to 2.");
              }
          }
      }
   }
   if(v_service.serviceID == 3) {
       for (var integer i:=0; i<sizeof(v_serviceParameterList); i:=i + 1) {
           if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 1) {
               if(v_serviceParameterList.parameterSet_list[i].
               parameter_list[0].name != "Protocol") {
                       setverdict(v_vct, "Invalid ParameterName (Protocol) was used for the " &
                                         "parameter serviceID equals to 3 and " &
                                         "'parameterSetID' equals to 1.");
                       break;
               }
               else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
               choice.stringValue != "ftp") {
                       setverdict(v_vct, "Invalid stringValue was used for the parameter " &
                                         "serviceID equals to 3 and " &
                                         "'parameterSetID' equals to 1.");
                       break;
               }
               else if(v_serviceParameterList.parameterSet_list[i].
               parameter_list[1].name != "Port") {
                       setverdict(v_vct, "Invalid ParameterName (Port) was used for the " &
                                         "parameter serviceID equals to 3 and " &
                                         "'parameterSetID' equals to 1.");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        break;
                    }
                    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[1].
                    choice.intValue != 20) {
                        setverdict(v_vct, "Invalid intValue was used for the parameter " &
                                          "serviceID equals to 3 and " &
                                          "'parameterSetID' equals to 1.");
                    }
                }
                if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 2) {
                    if(v_serviceParameterList.parameterSet_list[i].
                    parameter_list[0].name != "Protocol") {
                        setverdict(v_vct, "Invalid ParameterName (Protocol) was used for the " &
                                          "parameter serviceID equals to 3 and " &
                                          "'parameterSetID' equals to 2.");
                        break;
                    }
                    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
                    choice.stringValue != "ftp") {
                        setverdict(v_vct, "Invalid stringValue was used for the parameter " &
                                          "serviceID equals to 3 and " &
                                          "'parameterSetID' equals to 2.");
                        break;
                    }
                    else if(v_serviceParameterList.parameterSet_list[i].
                    parameter_list[1].name != "Port") {
                        setverdict(v_vct, "Invalid ParameterName (Port) was used for the " &
                                          "parameter serviceID equals to 3 and " &
                                          "'parameterSetID' equals to 2.");
                        break;
                    }
                    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[1].
                    choice.intValue != 21) {
                        setverdict(v_vct, "Invalid intValue was used for the parameter " &
                                          "serviceID equals to 3 and " &
                                          "'parameterSetID' equals to 2.");
                    }
                }
                if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 3) {
                    if(v_serviceParameterList.parameterSet_list[i].
                    parameter_list[0].name != "Protocol") {
                        setverdict(v_vct, "Invalid ParameterName (Protocol) was used for the " &
                                          "parameter serviceID equals to 3 and " &
                                          "'parameterSetID' equals to 3.");
                        break;
                    }
                    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
                    choice.stringValue != "http") {
                        setverdict(v_vct, "Invalid stringValue was used for the parameter " &
                                          "serviceID equals to 3 and " &
                                          "'parameterSetID' equals to 3.");
                        break;
                    }
                    else if(v_serviceParameterList.parameterSet_list[i].
                    parameter_list[1].name != "Port") {
                        setverdict(v_vct, "Invalid ParameterName (Port) was used for the " &
                                          "parameter serviceID equals to 3 and " &
                                          "'parameterSetID' equals to 3.");
                        break;
                    }
                    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[1].
                    choice.intValue != 80) {
                        setverdict(v_vct, "Invalid intValue was used for the parameter " &
                                          "serviceID equals to 3 and " &
                                          "'parameterSetID' equals to 3.");
                    }
                }
                if(v_serviceParameterList.parameterSet_list[i].parameterSetID == 4) {
                    if(v_serviceParameterList.parameterSet_list[i].
                    parameter_list[0].name != "Protocol") {
                        setverdict(v_vct, "Invalid ParameterName (Protocol) was used for the " &
                                          "parameter serviceID equals to 3 and " &
                                          "'parameterSetID' equals to 4.");
                        break;
                    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[0].
                    choice.stringValue != "https") {
                            setverdict(v_vct, "Invalid stringValue was used for the parameter " &
                                              "serviceID equals to 3 and " &
                                              "'parameterSetID' equals to 4.");
                            break;
                    }
                    else if(v_serviceParameterList.parameterSet_list[i].
                    parameter_list[1].name != "Port") {
                            setverdict(v_vct, "Invalid ParameterName (Port) was used for the " &
                                              "parameter serviceID equals to 3 and " &
                                              "'parameterSetID' equals to 4.");
                            break;
                    }
                    else if (v_serviceParameterList.parameterSet_list[i].parameter_list[1].
                    choice.intValue != 443) {
                            setverdict(v_vct, "Invalid intValue was used for the parameter " &
                                                "serviceID equals to 3 and " &
                                                "'parameterSetID' equals to 4.");
                    }
                }
            }
        }

    }

    function f_checkServiceCategory(ServiceCategoryType v_serviceCategory,
                                    ServiceListType vc_serviceList) return boolean {

      for (var integer i:=0; i<sizeof(vc_serviceList.service_list); i:=i + 1) {
          if(v_serviceCategory == vc_serviceList.service_list[i].serviceType.serviceCategory){
              return true;
          }
      }
      setverdict(inconc,"Selected serviceCategory could not be found within serviceList.");
      return false;
    }
}
```

I.2.2.4 SECC functions for PaymentServiceSelection

```
module TestBehavior_SECC_PaymentServiceSelection {

    import from Pics_15118_2 all;
    import from Pixit_15118_2 all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Services_HAL_61851_1 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from XSDAUX all;
    import from Timer_15118_2 all;
    import from Templates_SECC_ServiceDiscovery all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from TestBehavior_SECC_ServiceDetail all;
    import from Templates_SECC_PaymentServiceSelection all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from ComponentsAndPorts all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Timer_15118 all;
    import from LibFunctions_15118_2 all;
    import from Templates_SECC_PaymentDetails all;

    function f_SECC_CMN_TB_VTB_PaymentServiceSelection_001(in verdicttype v_vct)
                                                           runs on SECC_Tester
                                                           return verdicttype {

        var SessionIDType sessionID := vc_SessionID;
        var charstring strMsg;
        var V2G_Message v_responseMessage;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var charstring v_message;
var ServiceIDType v_serviceID;
var XSDAUX.short v_parameterSetID;

if (PICS_CMN_CMN_IdentificationMode == pnC){
   if(PIXIT_SECC_CMN_CertificateHandling == none_) {
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(sessionID),
                                              md_CMN_BodyType_001(
                                              md_SECC_CMN_PaymentServiceSelectionReq_001(
                                              contract,1))));


   } else {
        if(PIXIT_SECC_CMN_CertificateHandling == certInstall) {
            if(not ispresent(vc_serviceList)) {
                setverdict(inconc, "No certificate service was " &
                                   "provided within ServiceList. " &
                                   "Certificate Installation cannot be tested.");
            } else {
                v_serviceID := 2;
                v_parameterSetID := 1;
            }

        } else {
            if(not ispresent(vc_serviceList)) {
                setverdict(inconc, "No certificate service was " &
                                   "provided within ServiceList. " &
                                   "Certificate Update cannot be tested.");
            } else {
                v_serviceID := 2;
                v_parameterSetID := 2;
            }
        }
        if(getverdict == pass) {
            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(sessionID),
                                                  md_CMN_BodyType_001(
                                                  md_SECC_CMN_PaymentServiceSelectionReq_003(
                                                  contract, v_serviceID, v_parameterSetID))));

        }
   }
}
else{
   tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(sessionID),
                                          md_CMN_BodyType_001(
                                          md_SECC_CMN_PaymentServiceSelectionReq_001(
                                          externalPayment,1))));

}
if(getverdict == pass) {
    alt {
     [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001 (
                                          md_SECC_CMN_PaymentServiceSelectionRes_001(
                                         m_CMN_OkResponseCodeList_001))))
                                          -> value v_responseMessage {
           tc_V2G_EVCC_Msg_Timer.stop;
           setverdict ( pass,"PaymentServiceSelectionRes message was correct.");
     }
     [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                    "was misleadingly " &
                                                    "terminated by the SUT.");
     [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                     {paymentServiceSelectionRes := ?});
     [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                  "but cannot be decoded.");
     [] a_SECC_Unexpected_Message_Content(v_vct, "Received " &
                                                  "PaymentServiceSelectionRes " &
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                        "message, but with invalid " &
                                                        "data content.");
          [] a_SECC_Timeout(v_vct);
          }
     }
     return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentServiceSelection_002() runs on SECC_Tester
                                                          return verdicttype {

    var PaymentOptionType selPaymentOption;
    var SessionIDType sessionID := vc_SessionID;
    var charstring strMsg;
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

     if ( PICS_CMN_CMN_IdentificationMode == pnC ){
       selPaymentOption := contract;
     }
     else {
       selPaymentOption := externalPayment;
     }

          f_SECC_setIsConfirmationFlagDC(true);
         f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                          vc_FrequencyOff);
         f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                          vc_DutyCycleOscOff);
         deactivate(vc_Default_IEC_61851_ListenerBehavior);

     tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
     tc_CMN_TCP_Connection_Termination_Timer.start(
     par_CMN_TCP_Connection_Termination_Timeout);

     pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

       alt
        {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        []    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                          fAILED_SequenceError)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                            "'fAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                         e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                       {serviceDiscoveryRes := ?});
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionRes " &
                                                   "message, but with invalid " &
                                                   "data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                        "message with responseCode 'fAILED_SequenceError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
       return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentServiceSelection_003() runs on SECC_Tester
                                                         return verdicttype {

    var PaymentOptionType selPaymentOption;
    var SessionIDType sessionID := vc_SessionID;
    var charstring strMsg;
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

    if ( PICS_CMN_CMN_IdentificationMode == pnC ){
      selPaymentOption := contract;
    }
    else {
      selPaymentOption := externalPayment;
    }

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(
                                 f_rnd_SessionID(1,429496)),
                                 md_CMN_BodyType_001 (
                                md_SECC_CMN_PaymentServiceSelectionReq_001(
                                selPaymentOption,1))));
      alt {
       []
       a_Shutdown_Oscillator(pass){
            isShutdownOSC := true;
            if(isClosedTCP == false) {repeat;}
       }
       [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(?),
                                         md_CMN_BodyType_001 (
                                         md_SECC_CMN_PaymentServiceSelectionRes_001(
                                         fAILED_UnknownSession)))) {
           tc_V2G_EVCC_Msg_Timer.stop;
           setverdict(pass,"PaymentServiceSelectionRes with responseCode " &
                           "'fAILED_UnknownSession' is successful.");

           pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                         e_getPortStatus));
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            isErrorMessage := true;
            repeat;
         }
         [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                       "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
         }
         [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                          {paymentServiceSelectionRes := ?});
         [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                     "but cannot be decoded.");
         [] a_SECC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionRes " &
                                                    "message, but with invalid " &
                                                    "data content.");
         [] a_SECC_Timeout(fail);
         [] a_SECC_TCP_Connection_Termination_Timeout() {
             v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                             "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
         }
      }
      if(not isErrorMessage) {
          v_charstring := v_charstring & "SUT has not sent a PaymentServiceSelectionRes " &
                          "message with responseCode 'fAILED_UnknownSession' " &
                          "or other message data content was not conform. ";
      }
      if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
         setverdict(fail,v_charstring);
      }
      return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentServiceSelection_004() runs on SECC_Tester
                                                         return verdicttype {

    var PaymentOptionType selPaymentOption;
    var charstring v_message;
    var ServiceIDType v_invalidServiceID := 65535;
    var boolean isInvalid := true;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

     if ( PICS_CMN_CMN_IdentificationMode == pnC ) {
       selPaymentOption := contract;
     }
     else {
       selPaymentOption := externalPayment;
     }

     if(ispresent(vc_serviceList)) {
         isInvalid := f_checkInvalidServiceID(v_invalidServiceID, vc_serviceList);
     }

          f_SECC_setIsConfirmationFlagDC(true);
         f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                          vc_FrequencyOff);
         f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                          vc_DutyCycleOscOff);
         deactivate(vc_Default_IEC_61851_ListenerBehavior);

     if(isInvalid) {
         tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
         tc_CMN_TCP_Connection_Termination_Timer.start(
         par_CMN_TCP_Connection_Termination_Timeout);

         pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                       md_SECC_CMN_PaymentServiceSelectionReq_003(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        selPaymentOption,v_invalidServiceID,omit))));
          alt
            {
            []
            a_Shutdown_Oscillator(pass){
                isShutdownOSC := true;
                if(isClosedTCP == false) {repeat;}
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                                  md_CMN_BodyType_001 (
                                                md_SECC_CMN_PaymentServiceSelectionRes_001(
                                                fAILED_ServiceSelectionInvalid)))) {
              tc_V2G_EVCC_Msg_Timer.stop;
              setverdict(pass,"PaymentServiceSelectionRes with responseCode " &
                              "'fAILED_ServiceSelectionInvalid' is successful.");

              pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
              isErrorMessage := true;
              repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {paymentServiceSelectionRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionRes " &
                                                        "message, but with invalid " &
                                                        "data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
          }
          if(not isErrorMessage) {
              v_charstring := v_charstring & "SUT has not sent a PaymentServiceSelectionRes " &
                              "message with responseCode 'fAILED_ServiceSelectionInvalid' " &
                              "or other message data content was not conform. ";
          }
          if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
             setverdict(fail,v_charstring);
          }
      }
      return getverdict;
   }

   function f_SECC_CMN_TB_VTB_PaymentServiceSelection_005() runs on SECC_Tester
                                                            return verdicttype {

       var PaymentOptionType selPaymentOption;
       var charstring v_message;
       var boolean isShutdownOSC := false;
       var boolean isClosedTCP := false;
       var boolean isErrorMessage := false;
       var charstring v_charstring := "";

        if (PICS_CMN_CMN_IdentificationMode == pnC) {
          selPaymentOption := contract;
        }
        else{
          selPaymentOption := externalPayment;
        }
        vc_ServiceId := 2;

            f_SECC_setIsConfirmationFlagDC(true);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
          f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                           vc_FrequencyOff);
          f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                           vc_DutyCycleOscOff);
          deactivate(vc_Default_IEC_61851_ListenerBehavior);

      tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
      tc_CMN_TCP_Connection_Termination_Timer.start(
      par_CMN_TCP_Connection_Termination_Timeout);

      pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001 (
                                  md_SECC_CMN_PaymentServiceSelectionReq_002(
                                  selPaymentOption,vc_ServiceId,omit))));
        alt
         {
         []
         a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
         }
         [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_SECC_CMN_PaymentServiceSelectionRes_001(
                                            fAILED_NoChargeServiceSelected)))) {
             tc_V2G_EVCC_Msg_Timer.stop;
             setverdict(pass,"PaymentServiceSelectionRes with responseCode " &
                            "'fAILED_NoChargeServiceSelected' is successful.");

             pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
             isErrorMessage := true;
             repeat;
         }
         [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                       "terminated by the SUT.") {
              isClosedTCP := true;
              tc_CMN_TCP_Connection_Termination_Timer.stop;
              if(isShutdownOSC == false) {
                 logWarn("SECC oscillator was not shut down before the " &
                         "TCP connection was terminated.");}
              pt_HAL_61851_Internal_Port.clear;
         }
         [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                       {paymentServiceSelectionRes := ?});
         [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                     "but cannot be decoded.");
         [] a_SECC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionRes " &
                                                    "message, but with invalid " &
                                                    "data content.");
         [] a_SECC_Timeout(fail);
         [] a_SECC_TCP_Connection_Termination_Timeout() {
             v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
         }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a PaymentServiceSelectionRes " &
                            "message with responseCode 'fAILED_NoChargeServiceSelected' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentServiceSelection_006() runs on SECC_Tester
                                                         return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var float v_timeShift := 1.0;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
                 v_timeShift);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                        "terminated by the SUT " &
                                        "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentServiceSelection_007() runs on SECC_Tester
                                                         return verdicttype {

    var PaymentOptionType selPaymentOption;
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

    if(sizeof(vc_paymentOptionList.paymentOption_list) == 1) {
        if(PICS_CMN_CMN_IdentificationMode == pnC) {
            if(vc_paymentOptionList.paymentOption_list[0] != contract) {
                setverdict(inconc, "Invalid payment option was selected.");
            } else{
                selPaymentOption := externalPayment;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
        }
        else {
            if(vc_paymentOptionList.paymentOption_list[0] != externalPayment) {
                setverdict(inconc, "Invalid payment option was selected.");
            } else {
                selPaymentOption := contract;
            }
        }
    } else {
        setverdict(inconc, "The SUT has provided more than one payment option.");
    }

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(getverdict == pass) {

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    vc_SessionID),
                                    md_CMN_BodyType_001 (
                                    md_SECC_CMN_PaymentServiceSelectionReq_001(
                                    selPaymentOption,1))));
          alt
           {
           []
           a_Shutdown_Oscillator(pass){
                isShutdownOSC := true;
                if(isClosedTCP == false) {repeat;}
           }
           [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_SECC_CMN_PaymentServiceSelectionRes_001(
                                          fAILED_PaymentSelectionInvalid)))) {
               tc_V2G_EVCC_Msg_Timer.stop;
               setverdict(pass,"PaymentServiceSelectionRes with responseCode " &
                              "'fAILED_PaymentSelectionInvalid' is successful.");

               pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
               isErrorMessage := true;
               repeat;
           }
           [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT.") {
                isClosedTCP := true;
                tc_CMN_TCP_Connection_Termination_Timer.stop;
                if(isShutdownOSC == false) {
                   logWarn("SECC oscillator was not shut down before the " &
                           "TCP connection was terminated.");}
                pt_HAL_61851_Internal_Port.clear;
           }
           [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {paymentServiceSelectionRes := ?});
           [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                       "but cannot be decoded.");
           [] a_SECC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionRes " &
                                                      "message, but with invalid " &
                                                      "data content.");
           [] a_SECC_Timeout(fail);
           [] a_SECC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                              "within 'par_CMN_TCP_Connection_Termination_Timeout'.";
           }
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a PaymentServiceSelectionRes " &
                            "message with responseCode 'fAILED_PaymentSelectionInvalid' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }
 }
```

I.2.2.5 SECC functions for PaymentDetails

```
module TestBehavior_SECC_PaymentDetails {

    import from Timer_15118_2 all;
    import from Pics_15118_2 all;
    import from Pixit_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_SECC_PaymentDetails all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Services_HAL_61851_1 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Templates_CMN_Security all;
    import from Services_Security all;
    import from LibFunctions_15118_2 all;
    import from urn_iso_15118_2_2013_X509Def language "XSD" all;
    import from Timer_15118 all;

    function f_SECC_CMN_TB_VTB_PaymentDetails_001(in verdicttype v_vct)
                                                  runs on SECC_Tester
                                                  return verdicttype {

        var charstring v_message;
        var V2G_Message v_responseMessage;

        if(not(isbound(vc_contractSignatureCertChain))) {

            var X509CertificateType v_leafCert :=
            fx_loadCertificate(PIXIT_CMN_CMN_PkiName, cc_crt_contract_leaf_valid);
            var X509CertificateType v_subCert2 :=
            fx_loadCertificate(PIXIT_CMN_CMN_PkiName, cc_crt_mo_sub2_valid);
            var X509CertificateType v_subCert1 :=
            fx_loadCertificate(PIXIT_CMN_CMN_PkiName, cc_crt_mo_sub1_valid);
            vc_contractSignatureCertChain := md_CertificateChainType_002(v_leafCert,
                                             v_subCert2, v_subCert1)
        }
        if(getverdict == pass) {
            var EMAIDType_1 vc_EMAID := fx_getEMAID(vc_contractSignatureCertChain.
                                                    x509Certificate);

            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentDetailsReq);
            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_SECC_CMN_PaymentDetailsReq_001(vc_EMAID,
                                        vc_contractSignatureCertChain))));

            alt {
                []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                                    md_CMN_BodyType_001(
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                md_SECC_CMN_PaymentDetailsRes_001(oK,?,?))))
                                                -> value v_responseMessage {

                vc_GenChallenge := v_responseMessage.body.bodyElement.
                                   paymentDetailsRes.genChallenge;
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"PaymentDetailsRes message was correct.");
            }
            [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                           "was misleadingly " &
                                                           "terminated by the SUT.");
            [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {paymentDetailsRes := ?});
            [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(v_vct, "Received PaymentDetailsRes message, " &
                                                        "but with invalid data content.");
            [] a_SECC_Timeout(v_vct);
        }
    }
    return getverdict;
  }

function f_SECC_CMN_TB_VTB_PaymentDetails_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                            fAILED_SequenceError)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                            "'fAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {serviceDiscoveryRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                   "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received PaymentDetailsRes message, " &
                                                  "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                            "message with responseCode 'fAILED_SequenceError' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
      }

    function f_SECC_CMN_TB_VTB_PaymentDetails_003() runs on SECC_Tester return verdicttype {

        var charstring v_message;
        var boolean isShutdownOSC := false;
        var boolean isClosedTCP := false;
        var boolean isErrorMessage := false;
        var charstring v_charstring := "";

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        if(not(isbound(vc_contractSignatureCertChain))) {

            var X509CertificateType v_leafCert :=
            fx_loadCertificate(PIXIT_CMN_CMN_PkiName, cc_crt_contract_leaf_valid);
            var X509CertificateType v_subCert2 :=
            fx_loadCertificate(PIXIT_CMN_CMN_PkiName, cc_crt_mo_sub2_valid);
            var X509CertificateType v_subCert1 :=
            fx_loadCertificate(PIXIT_CMN_CMN_PkiName, cc_crt_mo_sub1_valid);
            vc_contractSignatureCertChain := md_CertificateChainType_002(v_leafCert,
                                             v_subCert2, v_subCert1)
        }
        if(getverdict == pass) {
            var EMAIDType_1 vc_EMAID := fx_getEMAID(vc_contractSignatureCertChain.
                                                    x509Certificate);

            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentDetailsReq);
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(
                                          f_rnd_SessionID(1,429496)),
                                          md_CMN_BodyType_001(
                                          md_SECC_CMN_PaymentDetailsReq_001(vc_EMAID,
                                          vc_contractSignatureCertChain))));

            alt {
                []
                a_Shutdown_Oscillator(pass){
                     isShutdownOSC := true;
                     if(isClosedTCP == false) {repeat;}
                }
                []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(?),
                                                md_CMN_BodyType_001(
                                                mdw_SECC_CMN_PaymentDetailsRes_002(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        fAILED_UnknownSession)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"PaymentDetailsRes with responseCode " &
                               "'fAILED_UnknownSession' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                          e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                         {paymentDetailsRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received PaymentDetailsRes message, " &
                                                       "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a PaymentDetailsRes " &
                        "message with responseCode 'fAILED_UnknownSession' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentDetails_004() runs on SECC_Tester return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var float v_timeShift := 1.0;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
                 v_timeShift);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                        "terminated by the SUT " &
                                        "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            };
        }
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_PaymentDetails_005(in charstring v_certName,
                                              in charstring v_sub2CertName,
                                              in charstring v_sub1CertName,
                                              in template ResponseCodeType
                                              v_responseCode)
                                              runs on SECC_Tester
                                              return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var ResponseCodeType v_responseCodeReceived;
    var V2G_Message v_responseMessage;

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    var X509CertificateType v_leafCert :=
    f_getCertificate(vc_IdCertificateMap, v_certName);
    var X509CertificateType v_subCert2 :=
    f_getCertificate(vc_IdCertificateMap, v_sub2CertName);
    var X509CertificateType v_subCert1 :=
    f_getCertificate(vc_IdCertificateMap, v_sub1CertName);
    if(getverdict == pass) {
        var EMAIDType_1 vc_EMAID := fx_getEMAID(v_leafCert);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PaymentDetailsReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     md_SECC_CMN_PaymentDetailsReq_001(vc_EMAID,
                                     md_CertificateChainType_002(v_leafCert,
                                     v_subCert2, v_subCert1)))));
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            alt {
                []
                a_Shutdown_Oscillator(pass){
                     isShutdownOSC := true;
                     if(isClosedTCP == false) {repeat;}
                }
                []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(
                                                vc_SessionID),
                                                md_CMN_BodyType_001(
                                                mdw_SECC_CMN_PaymentDetailsRes_002(
                                                v_responseCode))))
                                                -> value v_responseMessage {
                    tc_V2G_EVCC_Msg_Timer.stop;
                    v_responseCodeReceived := v_responseMessage.body.bodyElement.
                                              paymentDetailsRes.responseCode;
                    setverdict(pass,"PaymentDetailsRes with responseCode '" &
                                    toString<ResponseCodeType>(v_responseCodeReceived) &
                                     "' is successful.");

                    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                                  e_getPortStatus));
                    isErrorMessage := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                             "terminated by the SUT.") {
                     isClosedTCP := true;
                     tc_CMN_TCP_Connection_Termination_Timer.stop;
                     if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                                "TCP connection was terminated.");}
                     pt_HAL_61851_Internal_Port.clear;
                }
                [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                                 {paymentDetailsRes := ?});
                [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                            "but cannot be decoded.");
                [] a_SECC_Unexpected_Message_Content(fail, "Received PaymentDetailsRes message, " &
                                                           "but with invalid data content.");
                [] a_SECC_Timeout(fail);
                [] a_SECC_TCP_Connection_Termination_Timeout() {
                    v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                    "within 'par_CMN_TCP_Connection_Termination_Timeout' " &
                                    "or other message data content was not conform. ";
                }
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a PaymentDetailsRes " &
                            "message with responseCode '" &
                            toString<ResponseCodeType>(v_responseCodeReceived) & "'. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }
}
```

I.2.2.6 SECC functions for Authorization

```
module TestBehavior_SECC_Authorization {

    import from Pixit_15118 all;
    import from Timer_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from LibFunctions_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Services_HAL_61851_1 all;
```

 © ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from ComponentsAndPorts all;
import from Templates_CMN_Authorization all;
import from Templates_CMN_ServiceDiscovery all;
import from Templates_CMN_V2GMessage all;
import from Templates_SECC_ServiceDiscovery all;
import from Pics_15118_2 all;
import from TTlibrary_Logging all;
import from Pics_15118 all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from Templates_CMN_Security all;
import from http_www_w3_org_2000_09_xmldsig all;
import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
import from Services_Security all;
import from urn_iso_15118_2_2013_X509Def all;
import from Pixit_15118_2 all;
import from Timer_15118 all;

// EIM
function f_SECC_CMN_TB_VTB_Authorization_001(in verdicttype v_vct)
                                             runs on SECC_Tester
                                             return verdicttype {

    var EVSEProcessingType v_eVSEprocessing := ongoing_WaitingForCustomerInteraction;
    var charstring v_message;
    var V2G_Message v_responseMessage;
    var integer v_cnt := 0;

        var float v_dutyCycle := f_SECC_getMeasDutyCycle();
        if(v_dutyCycle == vc_DutyCycleOscOff and PICS_CMN_CMN_ChargingMode == aC) {
            f_SECC_changeValidDutyCycleRange(par_AC_validNominalDutyCycleLowerBound,
                                             par_AC_validNominalDutyCycleUpperBound);
            vc_validDutyCycleLowerBound2 := par_AC_validNominalDutyCycleLowerBound;
            vc_validDutyCycleUpperBound2 := par_AC_validNominalDutyCycleUpperBound;
        }

    if((PICS_SECC_CMN_EIMDone == v2gAuthorization or not vc_testCaseSpecific)) {
        var boolean v_result := f_SECC_CMN_EIMIdentification();
    }

    while (v_eVSEprocessing == ongoing_WaitingForCustomerInteraction){
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(m_CMN_CMN_AuthorizationReq_001)));
        alt {
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_AuthorizationRes_001(
                                        oK, finished)))) {
                v_eVSEprocessing := finished;
                tc_V2G_EVCC_Msg_Timer.stop;
                tc_V2G_EVCC_Ongoing_Timer.stop;
                v_cnt := v_cnt + 1;

                setverdict(pass,"AuthorizationRes message was correct.");
                if(PICS_CMN_CMN_ChargingMode == aC) {
                    vc_validDutyCycleLowerBound2 := par_AC_validNominalDutyCycleLowerBound;
                    vc_validDutyCycleUpperBound2 := par_AC_validNominalDutyCycleUpperBound;
                }
                if((PICS_SECC_CMN_EIMDone == v2gAuthorization or not vc_testCaseSpecific)) {
                    f_SECC_setEimStatus(true);
                }
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_AuthorizationRes_001(
                                        oK, ?))))
                                        -> value v_responseMessage {

                if(tc_V2G_EVCC_Ongoing_Timer.running==false){
                     tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
                }
                v_cnt := v_cnt + 1;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                var EVSEProcessingType v_ongoingType := v_responseMessage.body.
                                                        bodyElement.authorizationRes.
                                                        eVSEProcessing;

                if(v_ongoingType != ongoing_WaitingForCustomerInteraction) {

                        setverdict(v_vct, "Invalid EVSEProcessingType detected. " &
                                          "'Ongoing_WaitingForCustomerInteraction' " &
                                          "should be used.");
                }
            }
            [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                           "was misleadingly " &
                                                           "terminated by the SUT."){
                v_eVSEprocessing := finished;
                break;
            }
            [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {authorizationRes := ?}) {
                v_eVSEprocessing := finished;
                break;
            }
            [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.") {
                v_eVSEprocessing := finished;
                break;
            }
            [] a_SECC_Unexpected_Message_Content(v_vct, "Received AuthorizationRes message, " &
                                                        "but with invalid data content.") {
                v_eVSEprocessing := finished;
                break;
            }
            [] a_SECC_Timeout(v_vct){
                v_eVSEprocessing := finished;
                break;
            }
            []tc_V2G_EVCC_Ongoing_Timer.timeout{
                setverdict(v_vct,"V2G_EVCC_Ongoing_Timer has expired. " &
                                 "AuthorizationRes failed, " &
                                 "EVSEProcessing := Finished was " &
                                 "not sent in time.");
                v_eVSEprocessing := finished;
                break;
            }
        }
    }
    return getverdict;
}

//  PnC
function f_SECC_CMN_TB_VTB_Authorization_002(in verdicttype v_vct)
                                             runs on SECC_Tester
                                             return verdicttype {

    var EVSEProcessingType v_eVSEprocessing := ongoing;
    var charstring v_message;
    var integer v_cnt := 1;
    var charstring v_privateKey;
    var V2G_Message v_responseMessage;

        var float v_dutyCycle := f_SECC_getMeasDutyCycle();
        if(v_dutyCycle == vc_DutyCycleOscOff and PICS_CMN_CMN_ChargingMode == aC) {
            f_SECC_changeValidDutyCycleRange(par_AC_validNominalDutyCycleLowerBound,
                                             par_AC_validNominalDutyCycleUpperBound);
            vc_validDutyCycleLowerBound2 := par_AC_validNominalDutyCycleLowerBound;
            vc_validDutyCycleUpperBound2 := par_AC_validNominalDutyCycleUpperBound;
        }

    while (v_eVSEprocessing == ongoing){
        if(v_cnt == 1) {
            var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
                                                    md_CMN_CMN_AuthorizationReq_002(
                                                    "aReq", vc_GenChallenge));
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(isbound(vc_privateKey)) {
            v_privateKey := vc_privateKey;
        } else {
            v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
        }

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("aReq",
                                    f_generateDigestFromBodyType(
                                    myAuthorizationBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_003(
                                      vc_SessionID, aSignature),
                                      myAuthorizationBodyType));
    }
    else {
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      m_CMN_CMN_AuthorizationReq_001)));
    }
    v_cnt := v_cnt + 1;
    alt {
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_CMN_CMN_AuthorizationRes_001(
                                         oK, finished)))) {
            v_eVSEprocessing := finished;

            if(PICS_CMN_CMN_ChargingMode == aC) {
                vc_validDutyCycleLowerBound2 := par_AC_validNominalDutyCycleLowerBound;
                vc_validDutyCycleUpperBound2 := par_AC_validNominalDutyCycleUpperBound;
            }
            tc_V2G_EVCC_Msg_Timer.stop;
            tc_V2G_EVCC_Ongoing_Timer.stop;
            if(PICS_CMN_CMN_IEC61851) {
                f_SECC_setEimStatus(true);
            }
            setverdict(pass,"AuthorizationRes message was correct.");
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_CMN_CMN_AuthorizationRes_001(
                                         oK, ?)))) -> value v_responseMessage {

            if(tc_V2G_EVCC_Ongoing_Timer.running==false){
                 tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
            }

            var EVSEProcessingType v_ongoingType := v_responseMessage.body.
                                                    bodyElement.authorizationRes.
                                                    eVSEProcessing;

            if(v_ongoingType != ongoing) {

                    setverdict(v_vct, "Invalid EVSEProcessingType detected. " &
                                      "'Ongoing' should be used.");
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS " &
                                                       "connection was misleadingly" &
                                                       " terminated by the SUT."){
            v_eVSEprocessing := finished;
            break;
        }
        [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {authorizationRes := ?}) {
            v_eVSEprocessing := finished;
            break;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.") {
                v_eVSEprocessing := finished;
                break;
            }
            [] a_SECC_Unexpected_Message_Content(v_vct, "Received AuthorizationRes message, " &
                                                        "but with invalid data content.") {
                v_eVSEprocessing := finished;
                break;
            }
            [] a_SECC_Timeout(v_vct){
                v_eVSEprocessing := finished;
                break;
            }
            []tc_V2G_EVCC_Ongoing_Timer.timeout{
                setverdict(v_vct,"V2G_EVCC_Ongoing_Timer has expired. " &
                                 "AuthorizationRes failed, " &
                                 "EVSEProcessing := Finished was not sent in time.");
                v_eVSEprocessing := finished;
                break;
            }
        }
    }
    return getverdict;
}

//    EIM & PnC
function f_SECC_CMN_TB_VTB_Authorization_003() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
            isShutdownOSC := true;
            if(isClosedTCP == false) {repeat;}
        }
        []    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                          fAILED_SequenceError)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                            "'fAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                tc_CMN_TCP_Connection_Termination_Timer.stop;
                if(isShutdownOSC == false) {
                   logWarn("SECC oscillator was not shut down before the " &
                           "TCP connection was terminated.");}
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                             {serviceDiscoveryRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                   "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
                                                  "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                            "message with responseCode 'fAILED_SequenceError' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    //    EIM
    function f_SECC_CMN_TB_VTB_Authorization_004() runs on SECC_Tester return verdicttype {

        var charstring v_message;
        var boolean isShutdownOSC := false;
        var boolean isClosedTCP := false;
        var boolean isErrorMessage := false;
        var charstring v_charstring := "";

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    f_rnd_SessionID(1,429496)),
                                    md_CMN_BodyType_001(
                                    m_CMN_CMN_AuthorizationReq_001)));

        alt {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(?),
                                              md_CMN_BodyType_001(
                                              md_CMN_CMN_AuthorizationRes_001(
                                              fAILED_UnknownSession, ?)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"AuthorizationRes with responseCode " &
                                "'fAILED_UnknownSession' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {authorizationRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a AuthorizationRes " &
                        "message with responseCode 'fAILED_UnknownSession' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

//    PnC
function f_SECC_CMN_TB_VTB_Authorization_005() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var charstring v_privateKey;

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
                                            md_CMN_CMN_AuthorizationReq_002(
                                            "aReq", vc_GenChallenge));

    if(isbound(vc_privateKey)) {
        v_privateKey := vc_privateKey;
    } else {
        v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
    }

    var Signature aSignature := md_CMN_Signature_001(
                                md_CMN_SignedInfoType_001("aReq",
                                f_generateDigestFromBodyType(
                                myAuthorizationBodyType)),
                                v_privateKey);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_003(
                                  f_rnd_SessionID(1,429496), aSignature),
                                  myAuthorizationBodyType));

    alt {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(?),
                                              md_CMN_BodyType_001(
                                              md_CMN_CMN_AuthorizationRes_001(
                                              fAILED_UnknownSession, ?)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"AuthorizationRes with responseCode " &
                               "'fAILED_UnknownSession' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {authorizationRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
                                                       "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a AuthorizationRes " &
                            "message with responseCode 'fAILED_UnknownSession' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    //    PnC
    function f_SECC_CMN_TB_VTB_Authorization_006() runs on SECC_Tester return verdicttype {

        var charstring v_message;
        var boolean isShutdownOSC := false;
        var boolean isClosedTCP := false;
        var boolean isErrorMessage := false;
        var charstring v_charstring := "";
        var charstring v_privateKey;

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        vc_GenChallenge := fx_generateGenChallenge();

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_CMN_AuthorizationReq_002(
                                        "aReq", vc_GenChallenge));

    if(isbound(vc_privateKey)) {
        v_privateKey := vc_privateKey;
    } else {
        v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
    }

    var Signature aSignature := md_CMN_Signature_001(
                                md_CMN_SignedInfoType_001("aReq",
                                f_generateDigestFromBodyType(
                                myAuthorizationBodyType)), v_privateKey);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_003(
                                  vc_SessionID, aSignature),
                                  myAuthorizationBodyType));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(
                                        vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_AuthorizationRes_001(
                                        fAILED_ChallengeInvalid, ?)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"AuthorizationRes with responseCode " &
                            "'fAILED_ChallengeInvalid' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                       e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                       {authorizationRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a AuthorizationRes " &
                        "message with responseCode 'fAILED_ChallengeInvalid' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_Authorization_007() runs on SECC_Tester return verdicttype {

    timer t_wait;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var float v_timeShift := 1.0;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout - v_timeShift);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

//    PnC
function f_SECC_CMN_TB_VTB_Authorization_008() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
                                                 md_CMN_CMN_AuthorizationReq_002(
                                                 "aReq", vc_GenChallenge));

        // load wrong private key
        var charstring v_invalidPrivateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                              cc_crt_oem_leaf_valid);

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("aReq",
                                    f_generateDigestFromBodyType(myAuthorizationBodyType)),
                                    v_invalidPrivateKey);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_003(
                                      vc_SessionID, aSignature),
                                      myAuthorizationBodyType));

        alt {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(
                                                 vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_CMN_AuthorizationRes_001(
                                                 fAILED_SignatureError, ?)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"AuthorizationRes with responseCode " &
                                "'fAILED_SignatureError' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {authorizationRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
                                                       "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a AuthorizationRes " &
                            "message with responseCode 'fAILED_SignatureError' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
//    PnC
function f_SECC_CMN_TB_VTB_Authorization_009() runs on SECC_Tester return verdicttype {

    var boolean isShutdownOSC := false;
    var charstring v_privateKey;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var V2G_Message v_responseMessage;
    var ResponseCodeType v_responseCodeReceived;

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
                                            md_CMN_CMN_AuthorizationReq_002(
                                            "aReq", vc_GenChallenge));

    if(isbound(vc_privateKey)) {
        v_privateKey := vc_privateKey;
    } else {
        v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
    }

    var Signature aSignature := md_CMN_Signature_001(
                                md_CMN_SignedInfoType_001("aReq",
                                f_generateDigestFromBodyType(
                                myAuthorizationBodyType)),
                                v_privateKey);

    // change genChallenge
    vc_GenChallenge:=fx_generateGenChallenge();
    myAuthorizationBodyType := md_CMN_BodyType_001(
                               md_CMN_CMN_AuthorizationReq_002(
                               "aReq", vc_GenChallenge));

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_003(
                                  vc_SessionID, aSignature),
                                  myAuthorizationBodyType));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(
                                         vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_CMN_CMN_AuthorizationRes_001(
                                         m_CMN_failedResponseCodeList_004, ?))))
                                         -> value v_responseMessage {
            tc_V2G_EVCC_Msg_Timer.stop;
            v_responseCodeReceived := v_responseMessage.body.bodyElement.
                                      authorizationRes.responseCode;
            setverdict(pass,"AuthorizationRes message with responseCode '" &
                            toString<ResponseCodeType>(v_responseCodeReceived) &
                            "' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {authorizationRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a AuthorizationRes " &
                        "message with responseCode 'fAILED_SignatureError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

// EIM
function f_SECC_CMN_TB_VTB_Authorization_010() runs on SECC_Tester return verdicttype {

    var EVSEProcessingType v_eVSEprocessing := ongoing_WaitingForCustomerInteraction;
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var float v_timeShift := 1.0;
    var charstring v_charstring := "";

    while (v_eVSEprocessing == ongoing_WaitingForCustomerInteraction){
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(m_CMN_CMN_AuthorizationReq_001)));
        alt {
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_AuthorizationRes_001(
                                        oK, finished)))) {
                v_eVSEprocessing := finished;
                tc_V2G_EVCC_Msg_Timer.stop;
                tc_V2G_EVCC_Ongoing_Timer.stop;
                setverdict(inconc,"No EIM authorization was initiated before.");
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_AuthorizationRes_001(
                                        oK, ongoing_WaitingForCustomerInteraction)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                if(tc_V2G_SECC_Ongoing_Timer.running==false){
                     tc_V2G_SECC_Ongoing_Timer.start(par_V2G_SECC_Ongoing_Performance_Timeout -
                                                v_timeShift);
                }
            }
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS " &
                                                          "connection was misleadingly " &
                                                          "terminated by the SUT."){
                v_eVSEprocessing := finished;
                break;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                {authorizationRes := ?}) {
    v_eVSEprocessing := finished;
    break;
}
[isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                          "but cannot be decoded.") {
    v_eVSEprocessing := finished;
    break;
}
[] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
                                         "but with invalid data content.") {
    v_eVSEprocessing := finished;
    break;
}
[] a_SECC_Timeout(fail){
    v_eVSEprocessing := finished;
    break;
}
[]tc_V2G_SECC_Ongoing_Timer.timeout{

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    sleep(v_timeShift - par_CMN_Transmission_Delay);

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(
                                          vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_AuthorizationRes_001(
                                          fAILED, ?)))) {

             setverdict(pass,"AuthorizationRes with responseCode " &
                            "'fAILED' is successful.");

             pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                              e_getPortStatus));
             isErrorMessage := true;
             repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                   "but TCP connection was " &
                                                   ">>NOT<< terminated by " &
                                                   "the SUT");
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated " &
                            "the TCP connection within " &
                            "'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    v_charstring := v_charstring & "SUT has not sent a AuthorizationRes " &
                                    "message with responseCode 'fAILED' " &
                                    "or other message data content was not conform. ";
                }
                if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
                   setverdict(fail,v_charstring);
                }
                v_eVSEprocessing := finished;
                break;
            }
        }
    }
    return getverdict;
}

//  PnC
function f_SECC_CMN_TB_VTB_Authorization_011() runs on SECC_Tester return verdicttype {

    var EVSEProcessingType v_eVSEprocessing := ongoing;
    var charstring v_message;
    var integer v_cnt := 1;
    var charstring v_privateKey;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var float v_timeShift := 1.0;
    var charstring v_charstring := "";

    while (v_eVSEprocessing == ongoing){
        if(v_cnt == 1) {
            var BodyType myAuthorizationBodyType := md_CMN_BodyType_001(
                                                    md_CMN_CMN_AuthorizationReq_002(
                                                    "aReq", vc_GenChallenge));

            if(isbound(vc_privateKey)) {
                v_privateKey := vc_privateKey;
            } else {
                v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
            }

            var Signature aSignature := md_CMN_Signature_001(
                                        md_CMN_SignedInfoType_001("aReq",
                                        f_generateDigestFromBodyType(
                                        myAuthorizationBodyType)), v_privateKey);

            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_003(
                                          vc_SessionID, aSignature),
                                          myAuthorizationBodyType));
        }
        else {
            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          m_CMN_CMN_AuthorizationReq_001)));
        }
        v_cnt := v_cnt + 1;
        alt {
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_AuthorizationRes_001(
                                                oK, finished)))) {
                v_eVSEprocessing := finished;
                tc_V2G_EVCC_Msg_Timer.stop;
                tc_V2G_EVCC_Ongoing_Timer.stop;
                setverdict(inconc,"No authorization should be triggered by the SUT before.");
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_AuthorizationRes_001(
                                                oK, ongoing)))) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    tc_V2G_EVCC_Msg_Timer.stop;
    if(tc_V2G_SECC_Ongoing_Timer.running==false){
         tc_V2G_SECC_Ongoing_Timer.start(par_V2G_SECC_Ongoing_Performance_Timeout -
                                         v_timeShift);
    }
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS " &
                                              "connection was misleadingly" &
                                              " terminated by the SUT."){
    v_eVSEprocessing := finished;
    break;
}
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                 {authorizationRes := ?}) {
    v_eVSEprocessing := finished;
    break;
}
[isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                            "but cannot be decoded.") {
    v_eVSEprocessing := finished;
    break;
}
[] a_SECC_Unexpected_Message_Content(fail, "Received AuthorizationRes message, " &
                                           "but with invalid data content.") {
    v_eVSEprocessing := finished;
    break;
}
[] a_SECC_Timeout(fail){
    v_eVSEprocessing := finished;
    break;
}
[]tc_V2G_SECC_Ongoing_Timer.timeout{

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_SECC_Ongoing_Timer.start(v_timeShift -
                                    par_CMN_Transmission_Delay);

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                           md_CMN_MessageHeaderType_001(
                                           vc_SessionID),
                                           md_CMN_BodyType_001(
                                           md_CMN_CMN_AuthorizationRes_001(
                                           fAILED, ?)))) {

             setverdict(pass,"AuthorizationRes with responseCode " &
                            "'fAILED' is successful.");

             pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                              e_getPortStatus));
             isErrorMessage := true;
             repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                               md_CMN_MessageHeaderType_001(
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                                    vc_SessionID),
                                                                    md_CMN_BodyType_001(
                                                                    md_CMN_CMN_AuthorizationRes_001(
                                                                    oK, ongoing)))) {

                                tc_V2G_EVCC_Msg_Timer.stop;
                                tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq);
                                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                                  md_CMN_MessageHeaderType_001(
                                                                  vc_SessionID),
                                                                  md_CMN_BodyType_001(
                                                                  m_CMN_CMN_AuthorizationReq_001)));
                                repeat;
                            }
                            [] tc_V2G_SECC_Ongoing_Timer.timeout {
                                tc_CMN_TCP_Connection_Termination_Timer.start(
                                par_CMN_TCP_Connection_Termination_Timeout);
                                repeat;
                            }
                            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                                       "but TCP connection was " &
                                                                       ">>NOT<< terminated by " &
                                                                       "the SUT");
                            [] a_SECC_TCP_Connection_Termination_Timeout() {
                                v_charstring := v_charstring & "SUT has not terminated " &
                                                "the TCP connection within " &
                                                "'par_CMN_TCP_Connection_Termination_Timeout'. ";
                            }
                        }
                        if(not isErrorMessage) {
                            v_charstring := v_charstring & "SUT has not sent a AuthorizationRes " &
                                            "message with responseCode 'fAILED' " &
                                            "or other message data content was not conform. ";
                        }
                        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
                           setverdict(fail,v_charstring);
                        }
                        v_eVSEprocessing := finished;
                        break;
                    }
                }
            }
            return getverdict;
        }
    }
```

I.2.2.7 SECC functions for ChargeParameterDiscovery

```
module TestBehavior_SECC_ChargeParameterDiscovery {

    import from Timer_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from Templates_SECC_ChargeParameterDiscovery all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_CMN_ChargingStatus all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Pics_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from XSDAUX all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_Security all;
    import from http_www_w3_org_2000_09_xmldsig all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Services_Security all;
import from Pixit_15118_2 all;
import from Timer_15118 all;
import from Pixit_15118 all;
import from Timer_15118 all;

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(in verdicttype v_vct)
                                                        runs on SECC_Tester
                                                        return verdicttype {

  var EVSEProcessingType v_EVSEprocessing := ongoing;
  var V2G_Message v_responseMessage;
  var charstring v_message;
  var float v_pMax;
  var boolean isSignatureOk := true;
  var float v_processingDummy := 5.0;
  var integer v_cnt := 0;

  if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeAC,
                                        vc_supportedEnergyTransferMode)) {

      while (v_EVSEprocessing == ongoing){

          tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

          if(PICS_SECC_CMN_DepartureTime) {

            if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                  omit, vc_requestedEnergyTransferModeAC,
                                                  vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                                  vc_EVMaxCurrent, vc_EVMinCurrent))));
            } else {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                  par_CMN_MaxEntriesSAScheduleTuple,
                                                  vc_requestedEnergyTransferModeAC,
                                                  vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                                  vc_EVMaxCurrent, vc_EVMinCurrent))));
            }
          }
          else  {

            if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                  omit, vc_requestedEnergyTransferModeAC,
                                                  omit, vc_EAmount, vc_EVMaxVoltage,
                                                  vc_EVMaxCurrent, vc_EVMinCurrent))));
            } else {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                  par_CMN_MaxEntriesSAScheduleTuple,
                                                  vc_requestedEnergyTransferModeAC,
                                                  omit, vc_EAmount, vc_EVMaxVoltage,
                                                  vc_EVMaxCurrent, vc_EVMinCurrent))));
            }
          }

          alt {
              [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_002(vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  mdw_SECC_AC_ChargeParameterDiscoveryRes_001(
                                                  finished, md_CMN_AC_EVSEStatus_001(?,?,?),
                                                  *))))
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    -> value v_responseMessage {

 v_EVSEprocessing := finished;
 tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
 v_cnt := v_cnt + 1;
 vc_SASchedule_List := v_responseMessage.body.bodyElement.
                       chargeParameterDiscoveryRes.sASchedules.
                       sAScheduleList;
 vc_PMaxSchedule := v_responseMessage.body.bodyElement.
                    chargeParameterDiscoveryRes.sASchedules.sAScheduleList.
                    sAScheduleTuple_list[0].pMaxSchedule;
 vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
                         sAScheduleTupleID;
 vc_EVSENominalVoltage := v_responseMessage.body.bodyElement.
                          chargeParameterDiscoveryRes.eVSEChargeParameter.
                          aC_EVSEChargeParameter.eVSENominalVoltage;
 f_check_PhysicalValue_valueRange(cc_eVSENominalVoltage,
                                  vc_EVSENominalVoltage, v,
                                  cc_maximum_value_physicalValue_case2,
                                 v_vct);
 vc_EVSEMaxCurrent := v_responseMessage.body.bodyElement.
                      chargeParameterDiscoveryRes.eVSEChargeParameter.
                      aC_EVSEChargeParameter.eVSEMaxCurrent;
 f_check_PhysicalValue_valueRange(cc_eVSEMaxCurrent, vc_EVSEMaxCurrent, a,
                                  cc_maximum_value_physicalValue_case1,
                                 v_vct);
 tc_V2G_EVCC_Msg_Timer.stop;
 tc_V2G_EVCC_Ongoing_Timer.stop;
 if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
    eVSEChargeParameter.aC_EVSEChargeParameter.aC_EVSEStatus.
    eVSENotification != none_) {
    logWarn("The field 'eVSENotification' should be set to 'none_'.");
 }
 if(vc_requestedEnergyTransferModeAC == aC_three_phase_core) {
     v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
               f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent) * 3.0;
 }
 else {
    v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
              f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent);
 }

 f_checkACPMaxScheduleEntries(vc_SASchedule_List, v_pMax, v_vct);
 f_checkPMaxSchedulesStructure(vc_SASchedule_List,
                               cc_maximum_value_physicalValue_case4,
                              PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
                              v_vct);
 if(getverdict == pass) {
     f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime,
                                       vc_SASchedule_List, v_vct);
}
var boolean v_isSalesTariff := ispresent(vc_SASchedule_List.
                              sAScheduleTuple_list[0].salesTariff);

if(getverdict == pass and v_isSalesTariff) {
   f_checkSalesTariffStructure(vc_SASchedule_List,
   PICS_SECC_CMN_MaxEntriesSAScheduleTuple, v_vct);
}
if(getverdict == pass and v_isSalesTariff) {
   f_checkSalesTariffTimeIntervals(vc_DepartureTime,
                                   vc_SASchedule_List, v_vct);
}

if(PICS_CMN_CMN_IdentificationMode == pnC and v_isSalesTariff) {
   var idDigestValueMap resultIdDigestValueMap :=
   fx_getIdDigestMapFromScheduleListType(vc_SASchedule_List,
                    par_V2G_EVCC_Sequence_Performance_Time);
   if(lengthof(resultIdDigestValueMap) > 0) {
       var Signature authSignature := v_responseMessage.header.signature_;
       var BodyType chargeParameterResBodyType := v_responseMessage.body;

       f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                   cc_algorithm_method_2, cc_algorithm_method_3,
                                   v_vct);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if(getverdict == pass) {

                    isSignatureOk := f_verifySignature(
                                     chargeParameterResBodyType,
                                     authSignature,
                                     vc_contractSignatureCertChain.
                                     subCertificates.
                                     x509Certificate_list[0], v_vct);

                if(isSignatureOk) {
                    setverdict(pass, "Valid signature of the sales " &
                                     "tariff was received.");
                }
                else {
                    setverdict(v_vct, "Invalid signature of the sales " &
                                      "tariff was received.");
                }
            }
        } else {
          setverdict(inconc, "Sales Tariff signature verification " &
                             "could not be executed within defined " &
                             "performance time.");
        }
    }
    if(getverdict == pass) {
        setverdict(pass,"ChargeParameterDiscoveryRes message " &
                        "was correct.");
    }
    vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
    tc_CMN_ProcessingTime_Timer.stop;
}
[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
                                        md_CMN_AC_EVSEStatus_001(?,?,?)))))
                                        -> value v_responseMessage {

  if(tc_V2G_EVCC_Ongoing_Timer.running==false){
     tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
  }
  v_cnt := v_cnt + 1;
  if(v_responseMessage.body.bodyElement.
        chargeParameterDiscoveryRes.eVSEChargeParameter.aC_EVSEChargeParameter.
        aC_EVSEStatus.eVSENotification != none_) {
     logWarn("The field 'eVSENotification' should be set to 'none_'.");
  }
}
[] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                               "was misleadingly " &
                                               "terminated by the SUT.") {
   v_EVSEprocessing := finished;
   break;
}
[] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                 {chargeParameterDiscoveryRes := ?}) {
   v_EVSEprocessing := finished;
   break;
}
[isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                             "but cannot be decoded.") {
   v_EVSEprocessing := finished;
   break;
}
[] a_SECC_Unexpected_Message_Content(v_vct, "Received ChargeParameterDiscoveryRes " &
                                            "message, but with invalid " &
                                            "data content.") {
   v_EVSEprocessing := finished;
   break;
}
[] a_SECC_Timeout(v_vct) {
   v_EVSEprocessing := finished;
   break;
}
[] tc_V2G_EVCC_Ongoing_Timer.timeout{
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    v_EVSEprocessing := finished;
                    setverdict(v_vct,"EVSEProcessing = Finished was not sent in time.");
                    break;
                }
            }
        }
    }
    else {setverdict(inconc,"Requested energy transfer mode is not supported " &
                            "by the SECC.");}
    return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
                                                        return verdicttype {

   var charstring v_message;
   var boolean isShutdownOSC := false;
   var boolean isClosedTCP := false;
   var boolean isErrorMessage := false;
   var charstring v_charstring := "";

       f_SECC_setIsConfirmationFlagDC(true);
       f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                        vc_FrequencyOff);
       f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                        vc_DutyCycleOscOff);

   if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeAC,
                                         vc_supportedEnergyTransferMode)) {

       tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
       tc_CMN_TCP_Connection_Termination_Timer.start(
       par_CMN_TCP_Connection_Termination_Timeout);

       pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    f_rnd_SessionID(1,429496)),
                                    md_CMN_BodyType_001(
                                    md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                    omit, vc_requestedEnergyTransferModeAC, vc_DepartureTime,
                                    vc_EAmount, vc_EVMaxVoltage,
                                    vc_EVMaxCurrent, vc_EVMinCurrent))));

       alt {
           []
           a_Shutdown_Oscillator(pass){
                isShutdownOSC := true;
                if(isClosedTCP == false) {repeat;}
           }
           [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_002(?),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
                                          fAILED_UnknownSession)))) {

                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
                                "'fAILED_UnknownSession' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                          e_getPortStatus));
                isErrorMessage := true;
                repeat;
           }
           [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT.") {
                isClosedTCP := true;
                tc_CMN_TCP_Connection_Termination_Timer.stop;
                if(isShutdownOSC == false) {
                   logWarn("SECC oscillator was not shut down before the " &
                           "TCP connection was terminated.");}
                pt_HAL_61851_Internal_Port.clear;
           }
           [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {chargeParameterDiscoveryRes := ?});
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
          [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                      "but cannot be decoded.");
          [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                                        "message, but with invalid " &
                                                        "data content.");
          [] a_SECC_Timeout(fail);
          [] a_SECC_TCP_Connection_Termination_Timeout() {
              v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                              "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
          }
       }
       if(not isErrorMessage) {
           v_charstring := v_charstring & "SUT has not sent a ChargeParameterDiscoveryRes " &
                           "message with responseCode 'fAILED_UnknownSession' " &
                           "or other message data content was not conform. ";
       }
       if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
          setverdict(fail,v_charstring);
       }
   } else {setverdict(inconc,"Requested energy transfer mode is not supported by the SECC.");}
   return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_003() runs on SECC_Tester
                                                            return verdicttype  {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
       f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                        vc_FrequencyOff);
       f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                        vc_DutyCycleOscOff);

    vc_requestedEnergyTransferModeAC := dC_extended;
    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                omit, vc_requestedEnergyTransferModeAC,
                                vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                vc_EVMaxCurrent, vc_EVMinCurrent))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_002(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
                                        fAILED_WrongEnergyTransferMode)))) {
             tc_V2G_EVCC_Msg_Timer.stop;
             setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
                             "'fAILED_WrongEnergyTransferMode' is successful.");

             pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
             isErrorMessage := true;
             repeat;
       }
       [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                     "terminated by the SUT.") {
            isClosedTCP := true;
            tc_CMN_TCP_Connection_Termination_Timer.stop;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if(isShutdownOSC == false) {
               logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
            pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                         {chargeParameterDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                                   "message, but with invalid " &
                                                   "data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ChargeParameterDiscoveryRes " &
                        "message with responseCode 'fAILED_WrongEnergyTransferMode' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_004() runs on SECC_Tester
                                                            return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

          f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_SECC_AC_ChargeParameterDiscoveryReq_002(
                                      vc_requestedEnergyTransferModeAC))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_002(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
                                                fAILED_WrongChargeParameter)))) {

             tc_V2G_EVCC_Msg_Timer.stop;
             setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
                             "'fAILED_WrongChargeParameter' is successful.");

             pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                                e_getPortStatus));
             isErrorMessage := true;
             repeat;
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                         {chargeParameterDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                                "message, but with invalid " &
                                                "data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ChargeParameterDiscoveryRes " &
                        "message with responseCode 'fAILED_WrongChargeParameter' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_005() runs on SECC_Tester
                                                        return verdicttype  {

  var EVSEProcessingType v_EVSEprocessing := ongoing;
  var V2G_Message v_responseMessage;
  var charstring v_message;
  var float v_pMax;
  var boolean isSignatureOk := true;

  if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeAC,
                                        vc_supportedEnergyTransferMode)) {

      while (v_EVSEprocessing == ongoing){

          tc_V2G_EVCC_Msg_Timer.start(
          par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

          if(PICS_SECC_CMN_DepartureTime) {

           if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
               pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                        omit, vc_requestedEnergyTransferModeAC,
                                        vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                        vc_EVMaxCurrent, vc_EVMinCurrent))));
           } else {
               pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                        par_CMN_MaxEntriesSAScheduleTuple,
                                        vc_requestedEnergyTransferModeAC,
                                        vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                        vc_EVMaxCurrent, vc_EVMinCurrent))));
           }
          }
          else  {

           if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
               pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                    omit, vc_requestedEnergyTransferModeAC,
                                    omit, vc_EAmount, vc_EVMaxVoltage,
                                    vc_EVMaxCurrent, vc_EVMinCurrent))));
 } else {
     pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                    par_CMN_MaxEntriesSAScheduleTuple,
                                    vc_requestedEnergyTransferModeAC,
                                    omit, vc_EAmount, vc_EVMaxVoltage,
                                    vc_EVMaxCurrent, vc_EVMinCurrent))));
 }
}

alt {
    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                    mdw_SECC_AC_ChargeParameterDiscoveryRes_001(
                                    finished, md_CMN_AC_EVSEStatus_001(?,?,?),
                                    *))))
                                    -> value v_responseMessage {

          v_EVSEprocessing := finished;
          vc_SASchedule_List := v_responseMessage.body.bodyElement.
                                chargeParameterDiscoveryRes.sASchedules.
                                sAScheduleList;
          vc_PMaxSchedule := v_responseMessage.body.bodyElement.
                             chargeParameterDiscoveryRes.sASchedules.
                             sAScheduleList.sAScheduleTuple_list[0].
                             pMaxSchedule;
          vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
                                  sAScheduleTupleID;
          vc_EVSENominalVoltage := v_responseMessage.body.bodyElement.
                                   chargeParameterDiscoveryRes.
                                   eVSEChargeParameter.aC_EVSEChargeParameter.
                                   eVSENominalVoltage;
          f_check_PhysicalValue_valueRange(cc_eVSENominalVoltage,
                                           vc_EVSENominalVoltage, v,
                                           cc_maximum_value_physicalValue_case2,
                                           fail);
          vc_EVSEMaxCurrent := v_responseMessage.body.bodyElement.
                               chargeParameterDiscoveryRes.eVSEChargeParameter.
                               aC_EVSEChargeParameter.eVSEMaxCurrent;
          f_check_PhysicalValue_valueRange(cc_eVSEMaxCurrent, vc_EVSEMaxCurrent, a,
                                           cc_maximum_value_physicalValue_case1,
                                           fail);
          tc_V2G_EVCC_Msg_Timer.stop;
          tc_V2G_EVCC_Ongoing_Timer.stop;
          if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
             eVSEChargeParameter.aC_EVSEChargeParameter.aC_EVSEStatus.
             eVSENotification != none_) {
             logWarn("The field 'eVSENotification' should be set to 'none_'.");
          }
          if(vc_requestedEnergyTransferModeAC == aC_three_phase_core) {
             v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
                       f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent) * 3.0;
          }
          else {
             v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
                       f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent);
          }
          f_checkACPMaxScheduleEntries(vc_SASchedule_List, v_pMax, fail);
          f_checkPMaxSchedulesStructure(vc_SASchedule_List,
                                        cc_maximum_value_physicalValue_case4,
                                        PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
                                        fail);
          if(getverdict == pass) {
             f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedule_List,
                                               fail);
          }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        if(getverdict == pass) {
                            if(f_compareToPreviousSAScheduleTuple(vc_SASchedule_List,
                                                                  vc_SAScheduleTupleId)) {
                                setverdict(pass,"ChargeParameterDiscoveryRes message was correct. " &
                                                "A valid SAScheduleTuple was received after " &
                                                "paused session.");
                            }
                            else {
                                setverdict(fail,"Invalid SAScheduleTuple was received after " &
                                                "paused session.");
                            }
                        }
                    }
                [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
                                                md_CMN_AC_EVSEStatus_001(?,?,?)))))
                                                -> value v_responseMessage {

                      if(tc_V2G_EVCC_Ongoing_Timer.running==false){
                         tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
                      }
                      if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
                           eVSEChargeParameter.aC_EVSEChargeParameter.aC_EVSEStatus.
                           eVSENotification != none_) {
                         logWarn("The field 'eVSENotification' should be set to 'none_'.");
                      }
                   }
                [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                             "was misleadingly " &
                                                             "terminated by the SUT.") {
                   v_EVSEprocessing := finished;
                   break;
                }
                [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                                {chargeParameterDiscoveryRes := ?}) {
                   v_EVSEprocessing := finished;
                   break;
                }
                [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                            "but cannot be decoded.") {
                   v_EVSEprocessing := finished;
                   break;
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                                           "message, but with invalid " &
                                                           "data content.") {
                   v_EVSEprocessing := finished;
                   break;
                }
                [] a_SECC_Timeout(fail) {
                   v_EVSEprocessing := finished;
                   break;
                }
                [] tc_V2G_EVCC_Ongoing_Timer.timeout{
                   v_EVSEprocessing := finished;
                   setverdict(fail,"EVSEProcessing = Finished was not sent in time.");
                   break;
                }
             }
          }
       }
    else {setverdict(inconc,"Requested energy transfer mode is not supported by the SECC.");}
    return getverdict;
  }

  function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(in verdicttype v_vct)
                                                          runs on SECC_Tester
                                                        return verdicttype {

    var EVSEProcessingType v_EVSEprocessing := ongoing;
    var V2G_Message v_responseMessage;
    var charstring v_message;
    var float v_pMax;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var boolean isSignatureOk := true;
var float v_processingDummy := 5.0;
var integer v_cnt := 0;

if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeAC,
                                      vc_supportedEnergyTransferMode)) {

    while (v_EVSEprocessing == ongoing){

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

        if(PICS_SECC_CMN_DepartureTime) {

          if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
              pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                 omit, vc_requestedEnergyTransferModeAC,
                                                 vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                                 vc_EVMaxCurrent, vc_EVMinCurrent))));
          } else {

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                               md_CMN_MessageHeaderType_001(vc_SessionID),
                                               md_CMN_BodyType_001(
                                               md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                               par_CMN_MaxEntriesSAScheduleTuple,
                                               vc_requestedEnergyTransferModeAC,
                                               vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                               vc_EVMaxCurrent, vc_EVMinCurrent))));
          }
        }
        else  {

            if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                                   md_CMN_BodyType_001(
                                                   md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                   omit, vc_requestedEnergyTransferModeAC,
                                                   omit, vc_EAmount, vc_EVMaxVoltage,
                                                   vc_EVMaxCurrent, vc_EVMinCurrent))));
            } else {

                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                                   md_CMN_BodyType_001(
                                                   md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                   par_CMN_MaxEntriesSAScheduleTuple,
                                                   vc_requestedEnergyTransferModeAC,
                                                   omit, vc_EAmount, vc_EVMaxVoltage,
                                                   vc_EVMaxCurrent, vc_EVMinCurrent))));
            }
        }

        alt {
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                     md_CMN_MessageHeaderType_002(vc_SessionID),
                                                     md_CMN_BodyType_001(
                                                     mdw_SECC_AC_ChargeParameterDiscoveryRes_001(
                                                     finished, md_CMN_AC_EVSEStatus_001(?,?,?),?))))
                                                     -> value v_responseMessage {

               v_EVSEprocessing := finished;
               tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
               v_cnt := v_cnt + 1;
               vc_SASchedule_List := v_responseMessage.body.bodyElement.
                                     chargeParameterDiscoveryRes.sASchedules.
                                     sAScheduleList;
               vc_PMaxSchedule := v_responseMessage.body.bodyElement.
                                  chargeParameterDiscoveryRes.sASchedules.sAScheduleList.
                                  sAScheduleTuple_list[0].pMaxSchedule;
               vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
                                       sAScheduleTupleID;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
vc_EVSENominalVoltage := v_responseMessage.body.bodyElement.
                         chargeParameterDiscoveryRes.eVSEChargeParameter.
                         aC_EVSEChargeParameter.eVSENominalVoltage;
f_check_PhysicalValue_valueRange(cc_eVSENominalVoltage,
                                   vc_EVSENominalVoltage, v,
                                cc_maximum_value_physicalValue_case2,
                                v_vct);
vc_EVSEMaxCurrent := v_responseMessage.body.bodyElement.
                     chargeParameterDiscoveryRes.eVSEChargeParameter.
                     aC_EVSEChargeParameter.eVSEMaxCurrent;
f_check_PhysicalValue_valueRange(cc_eVSEMaxCurrent, vc_EVSEMaxCurrent, a,
                                cc_maximum_value_physicalValue_case1,
                                v_vct);
tc_V2G_EVCC_Msg_Timer.stop;
tc_V2G_EVCC_Ongoing_Timer.stop;
if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
   eVSEChargeParameter.aC_EVSEChargeParameter.aC_EVSEStatus.
   eVSENotification != none_) {
   logWarn("The field 'eVSENotification' should be set to 'none_'.");
}
if(vc_requestedEnergyTransferModeAC == aC_three_phase_core) {
   v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
             f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent) * 3.0;
}
else {
   v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
             f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent);
}
    f_checkACPMaxScheduleEntries(vc_SASchedule_List,
                                   v_pMax, v_vct);
    if(getverdict == pass) {
        f_checkPMaxSchedulesStructure(vc_SASchedule_List,
        cc_maximum_value_physicalValue_case4,
        PICS_SECC_CMN_MaxEntriesSAScheduleTuple, v_vct);
    }
    if(getverdict == pass) {
       f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime,
                                         vc_SASchedule_List, v_vct);
    }
    if(getverdict == pass) {
          f_checkSalesTariffStructure(vc_SASchedule_List,
          PICS_SECC_CMN_MaxEntriesSAScheduleTuple, v_vct);
    }
    if(getverdict == pass) {
       f_checkSalesTariffTimeIntervals(vc_DepartureTime,
                                       vc_SASchedule_List, v_vct);
    }
if(PICS_CMN_CMN_IdentificationMode == pnC) {
   var idDigestValueMap resultIdDigestValueMap :=
   fx_getIdDigestMapFromScheduleListType(vc_SASchedule_List,
                                   par_V2G_EVCC_Sequence_Performance_Time);
   if(lengthof(resultIdDigestValueMap) > 0) {
       var Signature authSignature := v_responseMessage.header.signature_;
       var BodyType chargeParameterResBodyType := v_responseMessage.body;

       f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                   cc_algorithm_method_2, cc_algorithm_method_3,
                                   v_vct);

       if(getverdict == pass) {

               isSignatureOk := f_verifySignature(
                                chargeParameterResBodyType,
                                authSignature,
                                vc_contractSignatureCertChain.
                                subCertificates.
                                x509Certificate_list[0], v_vct);

           if(isSignatureOk) {
               setverdict(pass, "Valid signature of the sales " &
                                "tariff was received.");
           }
           else {
               setverdict(v_vct, "Invalid signature of the sales " &
                                 "tariff was received.");
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        }
                    }
                } else {
                    setverdict(inconc, "Sales Tariff signature verification " &
                                       "could not be executed within defined " &
                                       "performance time.");
                }
            }
            if(getverdict == pass) {
               setverdict(pass,"ChargeParameterDiscoveryRes message " &
                               "was correct.");
            }
            vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
            tc_CMN_ProcessingTime_Timer.stop;
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
                                          md_CMN_AC_EVSEStatus_001(?,?,?)))))
                                          -> value v_responseMessage {

          if(tc_V2G_EVCC_Ongoing_Timer.running==false){
             tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
          }
          v_cnt := v_cnt + 1;
          if(v_responseMessage.body.bodyElement.
             chargeParameterDiscoveryRes.eVSEChargeParameter.aC_EVSEChargeParameter.
             aC_EVSEStatus.eVSENotification != none_) {
             logWarn("The field 'eVSENotification' should be set to 'none_'.");
          }
        }
        [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                       "was misleadingly " &
                                                   "terminated by the SUT.") {
           v_EVSEprocessing := finished;
           break;
        }
        [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {chargeParameterDiscoveryRes := ?}) {
           v_EVSEprocessing := finished;
           break;
        }
        [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                 "but cannot be decoded.") {
           v_EVSEprocessing := finished;
           break;
        }
        [] a_SECC_Unexpected_Message_Content(v_vct, "Received ChargeParameterDiscoveryRes " &
                                                 "message, but with invalid " &
                                                 "data content.") {
           v_EVSEprocessing := finished;
           break;
        }
        [] a_SECC_Timeout(v_vct) {
           v_EVSEprocessing := finished;
           break;
        }
        [] tc_V2G_EVCC_Ongoing_Timer.timeout{
           v_EVSEprocessing := finished;
           setverdict(v_vct,"EVSEProcessing = Finished was not sent in time.");
           break;
        }
      }
    }
  }
  else {setverdict(inconc,"Requested energy transfer mode is not supported by the SECC.");}
  return getverdict;
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_007() runs on SECC_Tester
                                                 return verdicttype  {

  var EVSEProcessingType v_EVSEprocessing := ongoing;
  var V2G_Message v_responseMessage;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
var charstring v_message;
var float v_pMax;
var boolean isSignatureOk := true;

if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeAC,
                                      vc_supportedEnergyTransferMode)) {

    while (v_EVSEprocessing == ongoing){

        tc_V2G_EVCC_Msg_Timer.start(
        par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

        if(PICS_SECC_CMN_DepartureTime) {

          if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
              pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                      omit, vc_requestedEnergyTransferModeAC,
                                      vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                      vc_EVMaxCurrent, vc_EVMinCurrent))));
          } else {

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                    par_CMN_MaxEntriesSAScheduleTuple,
                                    vc_requestedEnergyTransferModeAC,
                                    vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                    vc_EVMaxCurrent, vc_EVMinCurrent))));
          }
        }
        else  {

            if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                      omit, vc_requestedEnergyTransferModeAC,
                                      omit, vc_EAmount, vc_EVMaxVoltage,
                                      vc_EVMaxCurrent, vc_EVMinCurrent))));
            } else {

                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                      par_CMN_MaxEntriesSAScheduleTuple,
                                      vc_requestedEnergyTransferModeAC,
                                      omit, vc_EAmount, vc_EVMaxVoltage,
                                      vc_EVMaxCurrent, vc_EVMinCurrent))));
            }
        }

        alt {
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        mdw_SECC_AC_ChargeParameterDiscoveryRes_001(
                                        finished,
                                        md_CMN_AC_EVSEStatus_001(?,?,?),?))))
                                        -> value v_responseMessage {

                v_EVSEprocessing := finished;
                vc_SASchedule_List := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.sASchedules.
                                      sAScheduleList;
                vc_PMaxSchedule := v_responseMessage.body.bodyElement.
                                   chargeParameterDiscoveryRes.sASchedules.
                                   sAScheduleList.sAScheduleTuple_list[0].
                                   pMaxSchedule;
                vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        sAScheduleTupleID;
vc_EVSENominalVoltage := v_responseMessage.body.bodyElement.
                         chargeParameterDiscoveryRes.
                         eVSEChargeParameter.aC_EVSEChargeParameter.
                         eVSENominalVoltage;
f_check_PhysicalValue_valueRange(cc_eVSENominalVoltage,
                                    vc_EVSENominalVoltage, v,
                                 cc_maximum_value_physicalValue_case2,
                                 fail);
vc_EVSEMaxCurrent := v_responseMessage.body.bodyElement.
                     chargeParameterDiscoveryRes.eVSEChargeParameter.
                     aC_EVSEChargeParameter.eVSEMaxCurrent;
f_check_PhysicalValue_valueRange(cc_eVSEMaxCurrent, vc_EVSEMaxCurrent, a,
                                 cc_maximum_value_physicalValue_case1,
                                 fail);
tc_V2G_EVCC_Msg_Timer.stop;
tc_V2G_EVCC_Ongoing_Timer.stop;
if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
   eVSEChargeParameter.aC_EVSEChargeParameter.aC_EVSEStatus.
   eVSENotification != none_) {
   logWarn("The field 'eVSENotification' should be set to 'none_'.");
}
if(vc_requestedEnergyTransferModeAC == aC_three_phase_core) {
   v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
             f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent) * 3.0;
}
else {
   v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
             f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent);
}
f_checkACPMaxScheduleEntries(vc_SASchedule_List, v_pMax, fail);
if(getverdict == pass) {
    f_checkPMaxSchedulesStructure(vc_SASchedule_List,
    cc_maximum_value_physicalValue_case4,
    PICS_SECC_CMN_MaxEntriesSAScheduleTuple, fail);
}
if(getverdict == pass) {
   f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedule_List,
                                     fail);
}
if(getverdict == pass) {
   f_checkSalesTariffStructure(vc_SASchedule_List,
   PICS_SECC_CMN_MaxEntriesSAScheduleTuple, fail);
}
if(getverdict == pass) {
   f_checkSalesTariffTimeIntervals(vc_DepartureTime, vc_SASchedule_List,
                                   fail);
}
if(PICS_CMN_CMN_IdentificationMode == pnC) {
   var idDigestValueMap resultIdDigestValueMap :=
   fx_getIdDigestMapFromScheduleListType(vc_SASchedule_List,
                                         par_V2G_EVCC_Sequence_Performance_Time);
   if(lengthof(resultIdDigestValueMap) > 0) {
       var Signature authSignature := v_responseMessage.header.signature_;
       var BodyType chargeParameterResBodyType := v_responseMessage.body;

       f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                   cc_algorithm_method_2,
                                   cc_algorithm_method_3,
                                   fail);

       if(getverdict == pass) {

               isSignatureOk := f_verifySignature(
                                chargeParameterResBodyType,
                                authSignature,
                                vc_contractSignatureCertChain.
                                subCertificates.
                                x509Certificate_list[0], fail);

           if(isSignatureOk) {
               setverdict(pass, "Valid signature of the sales tariff " &
                                "was received.");
           }
           else {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    setverdict(fail, "Invalid signature of the sales tariff " &
                                                     "was received.");
                                }
                            }
                        } else {
                            setverdict(inconc, "Sales Tariff signature verification " &
                                               "could not be executed within defined " &
                                               "performance time.");
                        }
                    }
                    if(getverdict == pass) {
                        if(f_compareToPreviousSAScheduleTuple(vc_SASchedule_List,
                                                              vc_SAScheduleTupleId)) {
                            setverdict(pass,"ChargeParameterDiscoveryRes message was correct. " &
                                            "A valid SAScheduleTuple was received after " &
                                            "paused session.");
                        }
                        else {
                            setverdict(fail,"Invalid SAScheduleTuple was received after " &
                                            "paused session.");
                        }
                    }
                }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
                                                 md_CMN_AC_EVSEStatus_001(?,?,?)))))
                                                 -> value v_responseMessage {

                if(tc_V2G_EVCC_Ongoing_Timer.running==false){
                   tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
                }
                if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
                   eVSEChargeParameter.aC_EVSEChargeParameter.aC_EVSEStatus.
                   eVSENotification != none_) {
                   logWarn("The field 'eVSENotification' should be set to 'none_'.");
                }
            }
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                          "was misleadingly " &
                                                       "terminated by the SUT.") {
               v_EVSEprocessing := finished;
               break;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                             {chargeParameterDiscoveryRes := ?}) {
               v_EVSEprocessing := finished;
               break;
            }
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.") {
               v_EVSEprocessing := finished;
               break;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                                       "message, but with invalid " &
                                                       "data content.") {
               v_EVSEprocessing := finished;
               break;
            }
            [] a_SECC_Timeout(fail) {
               v_EVSEprocessing := finished;
               break;
            }
            [] tc_V2G_EVCC_Ongoing_Timer.timeout{
               v_EVSEprocessing := finished;
               setverdict(fail,"EVSEProcessing = Finished was not sent in time.");
               break;
            }
        }
    }
}
else {setverdict(inconc,"Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}

function f_SECC_AC_TB_VTB_ChargeParameterDiscovery_008() runs on SECC_Tester
                                                         return verdicttype {

  var EVSEProcessingType v_EVSEprocessing := ongoing;
  var charstring v_message;
  var boolean isShutdownOSC := false;
  var boolean isClosedTCP := false;
  var boolean isErrorMessage := false;
  var float v_timeShift := 1.0;
  var charstring v_charstring := "";

  if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeAC,
                                        vc_supportedEnergyTransferMode)) {

      while (v_EVSEprocessing == ongoing){

          tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

          if(PICS_SECC_CMN_DepartureTime) {

            if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                omit, vc_requestedEnergyTransferModeAC,
                                                vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                                vc_EVMaxCurrent, vc_EVMinCurrent))));
            } else {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                par_CMN_MaxEntriesSAScheduleTuple,
                                                vc_requestedEnergyTransferModeAC,
                                                vc_DepartureTime, vc_EAmount, vc_EVMaxVoltage,
                                                vc_EVMaxCurrent, vc_EVMinCurrent))));
            }
          }
          else  {

            if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                omit, vc_requestedEnergyTransferModeAC,
                                                omit, vc_EAmount, vc_EVMaxVoltage,
                                                vc_EVMaxCurrent, vc_EVMinCurrent))));
            } else {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                par_CMN_MaxEntriesSAScheduleTuple,
                                                vc_requestedEnergyTransferModeAC,
                                                omit, vc_EAmount, vc_EVMaxVoltage,
                                                vc_EVMaxCurrent, vc_EVMinCurrent))));
            }
          }

          alt {
              [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_002(vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  mdw_SECC_AC_ChargeParameterDiscoveryRes_001(
                                                  finished, md_CMN_AC_EVSEStatus_001(?,?,?),
                                                  *)))) {

                  v_EVSEprocessing := finished;
                  setverdict(inconc,"EVSEProcessingType 'Finished' was not expected. " &
                                    "Please check PIXIT configuration.");
              }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
[] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
                                    md_CMN_AC_EVSEStatus_001(?,?,?))))) {
      tc_V2G_EVCC_Msg_Timer.stop;
      if(tc_V2G_SECC_Ongoing_Timer.running==false){
          tc_V2G_SECC_Ongoing_Timer.start(
          par_V2G_SECC_Ongoing_Performance_Timeout -
          v_timeShift);
      }
}
[] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                              "was misleadingly " &
                                            "terminated by the SUT.") {
   v_EVSEprocessing := finished;
   break;
}
[] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                {chargeParameterDiscoveryRes := ?}) {
   v_EVSEprocessing := finished;
   break;
}
[isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                        "but cannot be decoded.") {
   v_EVSEprocessing := finished;
   break;
}
[] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                        "message, but with invalid " &
                                        "data content.") {
   v_EVSEprocessing := finished;
   break;
}
[] a_SECC_Timeout(fail) {
   v_EVSEprocessing := finished;
   break;
}
[] tc_V2G_SECC_Ongoing_Timer.timeout{

     f_SECC_setIsConfirmationFlagDC(true);
     f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                      vc_FrequencyOff);
     f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                      vc_DutyCycleOscOff);
     deactivate(vc_Default_IEC_61851_ListenerBehavior);

 tc_V2G_SECC_Ongoing_Timer.start(v_timeShift -
                                 par_CMN_Transmission_Delay);

 alt {
     []
     a_Shutdown_Oscillator(pass){
           isShutdownOSC := true;
           if(isClosedTCP == false) {repeat;}
     }
     [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_002(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
                                        fAILED)))) {

          setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
                          "'fAILED' is successful.");

          pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                      e_getPortStatus));
          isErrorMessage := true;
          repeat;
     }
     [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                   "terminated by the SUT.") {
          isClosedTCP := true;
          tc_CMN_TCP_Connection_Termination_Timer.stop;
          if(isShutdownOSC == false) {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                            logWarn("SECC oscillator was not shut down before the " &
                                    "TCP connection was terminated.");}
                         pt_HAL_61851_Internal_Port.clear;
                    }
                    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(

md_CMN_MessageHeaderType_001(vc_SessionID),
                                                            md_CMN_BodyType_001(

mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
                                                            md_CMN_AC_EVSEStatus_001(?,?,?))))) {

                        tc_V2G_EVCC_Msg_Timer.stop;

tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

                        if(PICS_SECC_CMN_DepartureTime) {

                          if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                              pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(

md_CMN_MessageHeaderType_001(vc_SessionID),
                                                            md_CMN_BodyType_001(

md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                            omit, vc_requestedEnergyTransferModeAC,
                                                            vc_DepartureTime,             vc_EAmount,
vc_EVMaxVoltage,
                                                            vc_EVMaxCurrent, vc_EVMinCurrent))));
                          } else {
                              pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(

md_CMN_MessageHeaderType_001(vc_SessionID),
                                                            md_CMN_BodyType_001(

md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                            par_CMN_MaxEntriesSAScheduleTuple,
                                                            vc_requestedEnergyTransferModeAC,
                                                            vc_DepartureTime,             vc_EAmount,
vc_EVMaxVoltage,
                                                            vc_EVMaxCurrent, vc_EVMinCurrent))));
                          }
                        }
                        else  {

                          if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                              pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(

md_CMN_MessageHeaderType_001(vc_SessionID),
                                                            md_CMN_BodyType_001(

md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                            omit, vc_requestedEnergyTransferModeAC,
                                                            omit, vc_EAmount, vc_EVMaxVoltage,
                                                            vc_EVMaxCurrent, vc_EVMinCurrent))));
                          } else {
                              pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(

md_CMN_MessageHeaderType_001(vc_SessionID),
                                                            md_CMN_BodyType_001(

md_SECC_AC_ChargeParameterDiscoveryReq_001(
                                                            par_CMN_MaxEntriesSAScheduleTuple,
                                                            vc_requestedEnergyTransferModeAC,
                                                            omit, vc_EAmount, vc_EVMaxVoltage,
                                                            vc_EVMaxCurrent, vc_EVMinCurrent))));
                          }
                        }
                        repeat;
                    }
                    [] tc_V2G_SECC_Ongoing_Timer.timeout {
                        tc_CMN_TCP_Connection_Termination_Timer.start(
                        par_CMN_TCP_Connection_Termination_Timeout);
                        repeat;
                    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                              "but TCP connection was " &
                                                              ">>NOT<< terminated by " &
                                                              "the SUT");
                    [] a_SECC_TCP_Connection_Termination_Timeout() {
                        v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                        "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
                    }
                }
                if(not isErrorMessage) {
                    v_charstring := v_charstring & "SUT has not sent a ChargeParameterDiscoveryRes " &
                                    "message with responseCode 'fAILED' " &
                                    "or other message data content was not conform. ";
                }
                if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
                   setverdict(fail,v_charstring);
                }
                v_EVSEprocessing := finished;
                break;
               }
            }
        }
    }
    else {setverdict(inconc,"Requested energy transfer mode is not supported " &
                            "by the SECC.");}
    return getverdict;
   }

   function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(in verdicttype v_vct)
                                                          runs on SECC_Tester
                                                          return verdicttype {

      var EVSEProcessingType v_EVSEprocessing := ongoing;
      var V2G_Message v_responseMessage;
      var charstring v_message;
      var boolean isSignatureOk := true;
      var boolean v_receivedOngoing := true;
      var float v_processingDummy := 5.0;
      var integer v_cnt := 0;

      if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing) {
         v_receivedOngoing := false;
      }

      if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeDC,
                                            vc_supportedEnergyTransferMode)) {

          while (v_EVSEprocessing == ongoing){
              log("-------------------------------------------------------------");
               log("EV maximum charging limits:");
              log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                  vc_EVMaximumCurrentLimit)));
              log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                  vc_EVMaximumVoltageLimit)));
              log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                  vc_EVMaximumPowerLimit)));
              log("-------------------------------------------------------------");

              tc_V2G_EVCC_Msg_Timer.start(
              par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

              if(PICS_SECC_CMN_DepartureTime) {

                 if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                     pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                omit, vc_requestedEnergyTransferModeDC,
                                                vc_DepartureTime, vc_DC_EVStatus,
                                                vc_EVMaximumCurrentLimit,
                                                vc_EVMaximumPowerLimit,
                                                vc_EVMaximumVoltageLimit))));
                 } else {
                     pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                    par_CMN_MaxEntriesSAScheduleTuple,
                                    vc_requestedEnergyTransferModeDC,
                                    vc_DepartureTime, vc_DC_EVStatus,
                                    vc_EVMaximumCurrentLimit,
                                    vc_EVMaximumPowerLimit,
                                    vc_EVMaximumVoltageLimit))));
   }
}
else{

   if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
       pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                    omit, vc_requestedEnergyTransferModeDC, omit,
                                    vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                    vc_EVMaximumPowerLimit,
                                    vc_EVMaximumVoltageLimit))));
   } else {
       pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                    par_CMN_MaxEntriesSAScheduleTuple,
                                    vc_requestedEnergyTransferModeDC, omit,
                                    vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                    vc_EVMaximumPowerLimit,
                                    vc_EVMaximumVoltageLimit))));
   }
}

alt {
    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                       md_CMN_MessageHeaderType_002(vc_SessionID),
                                       md_CMN_BodyType_001(
                                       mdw_SECC_DC_ChargeParameterDiscoveryRes_001(
                                       finished, none_, ?, *))))
                                       -> value v_responseMessage {

       v_EVSEprocessing := finished;
       tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
       v_cnt := v_cnt + 1;
       vc_SASchedule_List := v_responseMessage.body.bodyElement.
                             chargeParameterDiscoveryRes.sASchedules.
                             sAScheduleList;
       vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
                               sAScheduleTupleID;
       vc_PMaxSchedule := v_responseMessage.body.bodyElement.
                          chargeParameterDiscoveryRes.sASchedules.sAScheduleList.
                          sAScheduleTuple_list[0].pMaxSchedule;
       vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
                                     chargeParameterDiscoveryRes.
                                     eVSEChargeParameter.
                                     dC_EVSEChargeParameter.eVSEMaximumCurrentLimit;
       f_check_PhysicalValue_valueRange(cc_eVSEMaximumCurrentLimit,
                                       vc_EVSEMaximumCurrentLimit, a,
                                       cc_maximum_value_physicalValue_case1,
                                       v_vct);
       vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
                                     chargeParameterDiscoveryRes.
                                     eVSEChargeParameter.
                                     dC_EVSEChargeParameter.eVSEMaximumVoltageLimit;
       f_check_PhysicalValue_valueRange(cc_eVSEMaximumVoltageLimit,
                                       vc_EVSEMaximumVoltageLimit, v,
                                       cc_maximum_value_physicalValue_case2,
                                       v_vct);
       vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
                                   chargeParameterDiscoveryRes.
                                   eVSEChargeParameter.
                                   dC_EVSEChargeParameter.eVSEMaximumPowerLimit;
       f_check_PhysicalValue_valueRange(cc_eVSEMaximumPowerLimit,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                                vc_EVSEMaximumPowerLimit, w,
                                cc_maximum_value_physicalValue_case4,
                                v_vct);
vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
                              chargeParameterDiscoveryRes.
                              eVSEChargeParameter.
                              dC_EVSEChargeParameter.eVSEMinimumVoltageLimit;
f_check_PhysicalValue_valueRange(cc_eVSEMinimumVoltageLimit,
                                vc_EVSEMinimumVoltageLimit, v,
                                cc_maximum_value_physicalValue_case2,
                                v_vct);
vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.
                              chargeParameterDiscoveryRes.
                              eVSEChargeParameter.
                              dC_EVSEChargeParameter.eVSEMinimumCurrentLimit;
f_check_PhysicalValue_valueRange(cc_eVSEMinimumCurrentLimit,
                                vc_EVSEMinimumCurrentLimit, a,
                                cc_maximum_value_physicalValue_case1,
                                v_vct);
vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
                            chargeParameterDiscoveryRes.eVSEChargeParameter.
                            dC_EVSEChargeParameter.eVSEPeakCurrentRipple;
f_check_PhysicalValue_valueRange(cc_eVSEPeakCurrentRipple,
                                vc_EVSEPeakCurrentRipple, a,
                                cc_maximum_value_physicalValue_case1,
                                v_vct);
tc_V2G_EVCC_Msg_Timer.stop;
tc_V2G_EVCC_Ongoing_Timer.stop;
log("-------------------------------------------------------------");
log("EVSE maximum charging limits:");
log("EVSEMaximumCurrentLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumCurrentLimit)));
log("EVSEMaximumVoltageLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumVoltageLimit)));
log("EVSEMaximumPowerLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumPowerLimit)));
log("EVSEMinimumCurrentLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMinimumCurrentLimit)));
log("EVSEMinimumVoltageLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMinimumVoltageLimit)));
log("-------------------------------------------------------------");
    f_checkPMaxSchedulesStructure(vc_SASchedule_List,
                                  cc_maximum_value_physicalValue_case4,
                                  PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
                                  v_vct);
    f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedule_List,
                                      v_vct);

    var boolean v_isSalesTariff := ispresent(vc_SASchedule_List.
                                   sAScheduleTuple_list[0].salesTariff);

    if(getverdict == pass and v_isSalesTariff) {
       f_checkSalesTariffStructure(vc_SASchedule_List,
       PICS_SECC_CMN_MaxEntriesSAScheduleTuple, v_vct);
    }
    if(getverdict == pass and v_isSalesTariff) {
       f_checkSalesTariffTimeIntervals(vc_DepartureTime,
                                       vc_SASchedule_List, v_vct);
    }

    if(PICS_CMN_CMN_IdentificationMode == pnC and v_isSalesTariff) {
       var idDigestValueMap resultIdDigestValueMap :=
       fx_getIdDigestMapFromScheduleListType(vc_SASchedule_List,
                        par_V2G_EVCC_Sequence_Performance_Time);
       if(lengthof(resultIdDigestValueMap) > 0) {
           var Signature authSignature := v_responseMessage.header.signature_;
           var BodyType chargeParameterResBodyType := v_responseMessage.body;

           f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                       cc_algorithm_method_2,
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                cc_algorithm_method_3,
                                                v_vct);

                    if(getverdict == pass) {

                            isSignatureOk := f_verifySignature(
                                             chargeParameterResBodyType,
                                             authSignature,
                                             vc_contractSignatureCertChain.
                                             subCertificates.
                                             x509Certificate_list[0], v_vct);

                        if(isSignatureOk) {
                            setverdict(pass, "Valid signature of the sales " &
                                             "tariff was received.");
                        }
                        else {
                            setverdict(v_vct, "Invalid signature of the sales " &
                                              "tariff was received.");
                        }
                    }
                } else {
                  setverdict(inconc, "Sales Tariff signature verification " &
                                     "could not be executed within defined " &
                                     "performance time.");
                }
            }


        if(getverdict == pass) {
            setverdict(pass,"ChargeParameterDiscoveryRes message " &
                            "was correct.");
        }
        vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
        tc_CMN_ProcessingTime_Timer.stop;
    }
    []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_002(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
                                     none_, ?)))) -> value v_responseMessage {
      tc_V2G_EVCC_Msg_Timer.stop;
      v_receivedOngoing := true;
      v_cnt := v_cnt + 1;
      if(ispresent(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
                    eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
                    eVSEIsolationStatus)) {
          if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
             eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
             eVSEIsolationStatus != invalid) {
             logWarn("The field 'eVSEIsolationStatus' should be " &
                     "set to 'invalid'.");
          }
      }
      if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
           eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
           eVSEStatusCode != eVSE_Ready) {
         logWarn("The field 'eVSEStatusCode' should be set " &
                 "to 'eVSE_Ready'.");
      }
      if(tc_V2G_EVCC_Ongoing_Timer.running==false) {
           tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
      }
      vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
                                    chargeParameterDiscoveryRes.
                                    eVSEChargeParameter.
                                    dC_EVSEChargeParameter.eVSEMaximumCurrentLimit;
      f_check_PhysicalValue_valueRange(cc_eVSEMaximumCurrentLimit,
                                       vc_EVSEMaximumCurrentLimit, a,
                                     cc_maximum_value_physicalValue_case1,
                                     v_vct);
      vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
                                    chargeParameterDiscoveryRes.
                                    eVSEChargeParameter.
                                    dC_EVSEChargeParameter.eVSEMaximumVoltageLimit;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  f_check_PhysicalValue_valueRange(cc_eVSEMaximumVoltageLimit,
                                     vc_EVSEMaximumVoltageLimit, v,
                                  cc_maximum_value_physicalValue_case2,
                                  v_vct);
  vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
                              chargeParameterDiscoveryRes.
                              eVSEChargeParameter.
                              dC_EVSEChargeParameter.eVSEMaximumPowerLimit;
  f_check_PhysicalValue_valueRange(cc_eVSEMaximumPowerLimit,
                                     vc_EVSEMaximumPowerLimit, w,
                                  cc_maximum_value_physicalValue_case4,
                                  v_vct);
  vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
                                chargeParameterDiscoveryRes.
                                eVSEChargeParameter.
                                dC_EVSEChargeParameter.eVSEMinimumVoltageLimit;
  f_check_PhysicalValue_valueRange(cc_eVSEMinimumVoltageLimit,
                                     vc_EVSEMinimumVoltageLimit, v,
                                  cc_maximum_value_physicalValue_case2,
                                  v_vct);
  vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.
                                chargeParameterDiscoveryRes.
                                eVSEChargeParameter.
                                dC_EVSEChargeParameter.eVSEMinimumCurrentLimit;
  f_check_PhysicalValue_valueRange(cc_eVSEMinimumCurrentLimit,
                                     vc_EVSEMinimumCurrentLimit, a,
                                  cc_maximum_value_physicalValue_case1,
                                  v_vct);
  vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
                              chargeParameterDiscoveryRes.eVSEChargeParameter.
                              dC_EVSEChargeParameter.eVSEPeakCurrentRipple;
  f_check_PhysicalValue_valueRange(cc_eVSEPeakCurrentRipple,
                                     vc_EVSEPeakCurrentRipple, a,
                                  cc_maximum_value_physicalValue_case1,
                                  v_vct);
  log("-------------------------------------------------------------");
  log("EVSE maximum charging limits:");
  log("EVSEMaximumCurrentLimit: " & toString<float>(
  f_CMN_PhysicalValue_GetValue(
       vc_EVSEMaximumCurrentLimit)));
  log("EVSEMaximumVoltageLimit: " & toString<float>(
  f_CMN_PhysicalValue_GetValue(
       vc_EVSEMaximumVoltageLimit)));
  log("EVSEMaximumPowerLimit: " & toString<float>(
  f_CMN_PhysicalValue_GetValue(
       vc_EVSEMaximumPowerLimit)));
  log("EVSEMinimumCurrentLimit: " & toString<float>(
  f_CMN_PhysicalValue_GetValue(
       vc_EVSEMinimumCurrentLimit)));
  log("EVSEMinimumVoltageLimit: " & toString<float>(
  f_CMN_PhysicalValue_GetValue(
       vc_EVSEMinimumVoltageLimit)));
  log("-------------------------------------------------------------");

  setverdict(pass,"ChargeParameterDiscoveryRes message (Ongoing) " &
                  "was correct.");
}
[] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                               "was misleadingly " &
                                               "terminated by the SUT.") {
   v_EVSEprocessing := finished;
   break; }
[] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                 {chargeParameterDiscoveryRes := ?}) {
   v_EVSEprocessing := finished;
   break;
}
[isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                             "but cannot be decoded.") {
   v_EVSEprocessing := finished;
   break;
}
[] a_SECC_Unexpected_Message_Content(v_vct, "Received ChargeParameterDiscoveryRes " &
                                            "message, but with invalid " &
                                            "data content.") {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                v_EVSEprocessing := finished;
                break;
             }
             [] a_SECC_Timeout(v_vct) {
                v_EVSEprocessing := finished;
                break; }
             [] tc_V2G_EVCC_Ongoing_Timer.timeout{
                v_EVSEprocessing := finished;
                setverdict(v_vct,"EVSEProcessing = Finished was not sent in time.");
                break;
             }
          }
       }
       if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing and
          not v_receivedOngoing) {
          setverdict(inconc, "The SUT did not repeat the ChargeParameter" &
                             " message sequence.");
       }
    }
    else {setverdict(inconc,"Requested energy transfer mode is not supported by the SECC.");}
    return getverdict;
  }

    function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
                                                             return verdicttype {

       var charstring v_message;
       var boolean isShutdownOSC := false;
       var boolean isClosedTCP := false;
       var boolean isErrorMessage := false;
       var charstring v_charstring := "";

           f_SECC_setIsConfirmationFlagDC(true);
           f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                            vc_FrequencyOff);
           f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                            vc_DutyCycleOscOff);

       if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeDC,
                                             vc_supportedEnergyTransferMode)) {
           tc_V2G_EVCC_Msg_Timer.start(
           par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
           tc_CMN_TCP_Connection_Termination_Timer.start(
           par_CMN_TCP_Connection_Termination_Timeout);

           pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(
                                            f_rnd_SessionID(1,429496)),
                                            md_CMN_BodyType_001(
                                            md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                            omit, vc_requestedEnergyTransferModeDC,
                                            vc_DepartureTime, vc_DC_EVStatus,
                                            vc_EVMaximumCurrentLimit, omit,
                                            vc_EVMaximumVoltageLimit))));

           alt {
               []
               a_Shutdown_Oscillator(pass){
                    isShutdownOSC := true;
                    if(isClosedTCP == false) {repeat;}
               }
               [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_002(?),
                                                md_CMN_BodyType_001(
                                                mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
                                                fAILED_UnknownSession)))) {

                    tc_V2G_EVCC_Msg_Timer.stop;
                    setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
                                   "'fAILED_UnknownSession' is successful.");

                    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                                e_getPortStatus));
                    isErrorMessage := true;
                    repeat;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {chargeParameterDiscoveryRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                  "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                                   "message, but with invalid " &
                                                   "data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a ChargeParameterDiscoveryRes " &
                            "message with responseCode 'fAILED_UnknownSession' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    } else {setverdict(inconc,"Requested energy transfer mode is not supported by the SECC.");}
    return getverdict;
}

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_003() runs on SECC_Tester
                                                         return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);

    vc_requestedEnergyTransferModeDC := aC_three_phase_core;
    tc_V2G_EVCC_Msg_Timer.start(
    par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                 omit, vc_requestedEnergyTransferModeDC, vc_DepartureTime,
                                 vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                 omit, vc_EVMaximumVoltageLimit))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_002(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
                                            fAILED_WrongEnergyTransferMode)))) {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
                            "'fAILED_WrongEnergyTransferMode' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {chargeParameterDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                              "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                                "message, but with invalid " &
                                                "data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ChargeParameterDiscoveryRes " &
                        "message with responseCode 'fAILED_WrongEnergyTransferMode' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_004() runs on SECC_Tester
                                                        return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

         f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                        vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                        vc_DutyCycleOscOff);

    tc_V2G_EVCC_Msg_Timer.start(
    par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_SECC_DC_ChargeParameterDiscoveryReq_002(
                                vc_requestedEnergyTransferModeDC))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
          [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_002(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
                                          fAILED_WrongChargeParameter)))) {

               tc_V2G_EVCC_Msg_Timer.stop;
               setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
                                 "'fAILED_WrongChargeParameter' is successful.");

               pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                          e_getPortStatus));
               isErrorMessage := true;
               repeat;
          }
          [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                        "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if(isShutdownOSC == false) {
                  logWarn("SECC oscillator was not shut down before the " &
                          "TCP connection was terminated.");}
               pt_HAL_61851_Internal_Port.clear;
          }
          [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                       {chargeParameterDiscoveryRes := ?});
          [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                      "but cannot be decoded.");
          [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                                     "message, but with invalid " &
                                                     "data content.");
          [] a_SECC_Timeout(fail);
          [] a_SECC_TCP_Connection_Termination_Timeout() {
              v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                              "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
          }
       }
       if(not isErrorMessage) {
           v_charstring := v_charstring & "SUT has not sent a ChargeParameterDiscoveryRes " &
                           "message with responseCode 'fAILED_WrongChargeParameter' " &
                           "or other message data content was not conform. ";
       }
       if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
          setverdict(fail,v_charstring);
       }
       return getverdict;
}

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_005() runs on SECC_Tester
                                                         return verdicttype {

   var EVSEProcessingType v_EVSEprocessing := ongoing;
   var V2G_Message v_responseMessage;
   var charstring v_message;
   var boolean isSignatureOk := true;
   var boolean v_receivedOngoing := true;

   if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing) {
      v_receivedOngoing := false;
   }

   if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeDC,
                                         vc_supportedEnergyTransferMode)) {

       while (v_EVSEprocessing == ongoing){
            log("-------------------------------------------------------------");
               log("EV maximum charging limits:");
            log("EVMaximumCurrentLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumCurrentLimit)));
            log("EVMaximumVoltageLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumVoltageLimit)));
            log("EVMaximumPowerLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    vc_EVMaximumPowerLimit)));
log("------------------------------------------------------------");

tc_V2G_EVCC_Msg_Timer.start(
par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

if(PICS_SECC_CMN_DepartureTime) {

   if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
       pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                    omit, vc_requestedEnergyTransferModeDC,
                                    vc_DepartureTime, vc_DC_EVStatus,
                                    vc_EVMaximumCurrentLimit,
                                    vc_EVMaximumPowerLimit,
                                    vc_EVMaximumVoltageLimit))));
   } else {
       pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                    par_CMN_MaxEntriesSAScheduleTuple,
                                    vc_requestedEnergyTransferModeDC,
                                    vc_DepartureTime, vc_DC_EVStatus,
                                    vc_EVMaximumCurrentLimit,
                                    vc_EVMaximumPowerLimit,
                                    vc_EVMaximumVoltageLimit))));
   }
}
else{

   if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
       pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                    omit, vc_requestedEnergyTransferModeDC, omit,
                                    vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                    vc_EVMaximumPowerLimit,
                                    vc_EVMaximumVoltageLimit))));
   } else {
       pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                    par_CMN_MaxEntriesSAScheduleTuple,
                                    vc_requestedEnergyTransferModeDC, omit,
                                    vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                    vc_EVMaximumPowerLimit,
                                    vc_EVMaximumVoltageLimit))));
   }
}

alt {
   [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_002(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        mdw_SECC_DC_ChargeParameterDiscoveryRes_001(
                                        finished, none_, ?, *))))
                                        -> value v_responseMessage {

          v_EVSEprocessing := finished;
          vc_SASchedule_List := v_responseMessage.body.bodyElement.
                                chargeParameterDiscoveryRes.sASchedules.
                                sAScheduleList;
          vc_PMaxSchedule := v_responseMessage.body.bodyElement.
                             chargeParameterDiscoveryRes.sASchedules.
                             sAScheduleList.
                             sAScheduleTuple_list[0].pMaxSchedule;
          vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
                                        chargeParameterDiscoveryRes.
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                                eVSEChargeParameter.
                                dC_EVSEChargeParameter.
                                eVSEMaximumCurrentLimit;
f_check_PhysicalValue_valueRange(cc_eVSEMaximumCurrentLimit,
                                   vc_EVSEMaximumCurrentLimit, a,
                                   cc_maximum_value_physicalValue_case1,
                                   fail);
vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
                                chargeParameterDiscoveryRes.
                                eVSEChargeParameter.
                                dC_EVSEChargeParameter.
                                eVSEMaximumVoltageLimit;
f_check_PhysicalValue_valueRange(cc_eVSEMaximumVoltageLimit,
                                   vc_EVSEMaximumVoltageLimit, v,
                                   cc_maximum_value_physicalValue_case2,
                                   fail);
vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
                              chargeParameterDiscoveryRes.
                              eVSEChargeParameter.
                              dC_EVSEChargeParameter.eVSEMaximumPowerLimit;
f_check_PhysicalValue_valueRange(cc_eVSEMaximumPowerLimit,
                                   vc_EVSEMaximumPowerLimit, w,
                                   cc_maximum_value_physicalValue_case4,
                                   fail);
vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
                                chargeParameterDiscoveryRes.
                                eVSEChargeParameter.
                                dC_EVSEChargeParameter.
                                eVSEMinimumVoltageLimit;
f_check_PhysicalValue_valueRange(cc_eVSEMinimumVoltageLimit,
                                   vc_EVSEMinimumVoltageLimit, v,
                                   cc_maximum_value_physicalValue_case2,
                                   fail);
vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.
                                chargeParameterDiscoveryRes.
                                eVSEChargeParameter.
                                dC_EVSEChargeParameter.
                                eVSEMinimumCurrentLimit;
f_check_PhysicalValue_valueRange(cc_eVSEMinimumCurrentLimit,
                                   vc_EVSEMinimumCurrentLimit, a,
                                   cc_maximum_value_physicalValue_case1,
                                   fail);
vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
                              chargeParameterDiscoveryRes.
                              eVSEChargeParameter.
                              dC_EVSEChargeParameter.
                              eVSEPeakCurrentRipple;
f_check_PhysicalValue_valueRange(cc_eVSEPeakCurrentRipple,
                                   vc_EVSEPeakCurrentRipple, a,
                                   cc_maximum_value_physicalValue_case1,
                                   fail);
tc_V2G_EVCC_Msg_Timer.stop;
tc_V2G_EVCC_Ongoing_Timer.stop;
log("-------------------------------------------------------------");
log("EVSE maximum charging limits:");
log("EVSEMaximumCurrentLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumCurrentLimit)));
log("EVSEMaximumVoltageLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumVoltageLimit)));
log("EVSEMaximumPowerLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumPowerLimit)));
log("EVSEMinimumCurrentLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMinimumCurrentLimit)));
log("EVSEMinimumVoltageLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMinimumVoltageLimit)));
log("-------------------------------------------------------------");
f_checkPMaxSchedulesStructure(vc_SASchedule_List,
                              cc_maximum_value_physicalValue_case4,
                              PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
                              fail);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedule_List,
                                              fail);
            if(getverdict == pass) {
                if(f_compareToPreviousSAScheduleTuple(vc_SASchedule_List,
                                                      vc_SAScheduleTupleId)) {
                    setverdict(pass,"ChargeParameterDiscoveryRes message " &
                                    "was correct. A valid SAScheduleTuple " &
                                    "was received after paused session.");
                  }
                else {
                    setverdict(fail,"Invalid SAScheduleTuple was received after " &
                                    "paused session.");
                }
            }
        }
[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                       md_CMN_MessageHeaderType_002(vc_SessionID),
                                       md_CMN_BodyType_001(
                                     mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
                                     none_, ?)))) -> value v_responseMessage {

        if(ispresent(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
                     eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
                     eVSEIsolationStatus)) {
            if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
               eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
               eVSEIsolationStatus != invalid) {
               logWarn("The field 'eVSEIsolationStatus' should be " &
                          "set to 'invalid'.");
            }
        }
        if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
           eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
           eVSEStatusCode != eVSE_Ready) {
           logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
        }
        if(tc_V2G_EVCC_Ongoing_Timer.running==false) {
             tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
        }
        vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.
                                      eVSEChargeParameter.
                                      dC_EVSEChargeParameter.
                                      eVSEMaximumCurrentLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMaximumCurrentLimit,
                                         vc_EVSEMaximumCurrentLimit, a,
                                         cc_maximum_value_physicalValue_case1,
                                         fail);
        vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.
                                      eVSEChargeParameter.
                                      dC_EVSEChargeParameter.
                                      eVSEMaximumVoltageLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMaximumVoltageLimit,
                                         vc_EVSEMaximumVoltageLimit, v,
                                         cc_maximum_value_physicalValue_case2,
                                         fail);
        vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
                                    chargeParameterDiscoveryRes.
                                    eVSEChargeParameter.
                                    dC_EVSEChargeParameter.
                                    eVSEMaximumPowerLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMaximumPowerLimit,
                                         vc_EVSEMaximumPowerLimit, w,
                                         cc_maximum_value_physicalValue_case4,
                                         fail);
        vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.
                                      eVSEChargeParameter.
                                      dC_EVSEChargeParameter.
                                      eVSEMinimumVoltageLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMinimumVoltageLimit,
                                         vc_EVSEMinimumVoltageLimit, v,
                                         cc_maximum_value_physicalValue_case2,
                                         fail);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.
                                              chargeParameterDiscoveryRes.
                                              eVSEChargeParameter.
                                              dC_EVSEChargeParameter.
                                              eVSEMinimumCurrentLimit;
                f_check_PhysicalValue_valueRange(cc_eVSEMinimumCurrentLimit,
                                                 vc_EVSEMinimumCurrentLimit, a,
                                                 cc_maximum_value_physicalValue_case1,
                                                 fail);
                vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
                                            chargeParameterDiscoveryRes.eVSEChargeParameter.
                                            dC_EVSEChargeParameter.eVSEPeakCurrentRipple;
                f_check_PhysicalValue_valueRange(cc_eVSEPeakCurrentRipple ,
                                                 vc_EVSEPeakCurrentRipple, a,
                                                 cc_maximum_value_physicalValue_case1,
                                                 fail);
                log("------------------------------------------------------------");
                log("EVSE maximum charging limits:");
                log("EVSEMaximumCurrentLimit: " & toString<float>(
                f_CMN_PhysicalValue_GetValue(
                     vc_EVSEMaximumCurrentLimit)));
                log("EVSEMaximumVoltageLimit: " & toString<float>(
                f_CMN_PhysicalValue_GetValue(
                     vc_EVSEMaximumVoltageLimit)));
                log("EVSEMaximumPowerLimit: " & toString<float>(
                f_CMN_PhysicalValue_GetValue(
                     vc_EVSEMaximumPowerLimit)));
                log("EVSEMinimumCurrentLimit: " & toString<float>(
                f_CMN_PhysicalValue_GetValue(
                     vc_EVSEMinimumCurrentLimit)));
                log("EVSEMinimumVoltageLimit: " & toString<float>(
                f_CMN_PhysicalValue_GetValue(
                     vc_EVSEMinimumVoltageLimit)));
                log("------------------------------------------------------------");

                setverdict(pass,"ChargeParameterDiscoveryRes message (Ongoing) " &
                                "was correct.");
             }
             [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                        "was misleadingly " &
                                                        "terminated by the SUT.") {
                v_EVSEprocessing := finished;
                break; }
             [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                              {chargeParameterDiscoveryRes := ?}) {
                v_EVSEprocessing := finished;
                break;
             }
             [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                   "but cannot be decoded.") {
                v_EVSEprocessing := finished;
                break;
             }
             [] a_SECC_Unexpected_Message_Content(fail, "Received " &
                                                   "ChargeParameterDiscoveryRes " &
                                                   "message, but with invalid " &
                                                   "data content.") {
                v_EVSEprocessing := finished;
                break;
             }
             [] a_SECC_Timeout(fail) {
                v_EVSEprocessing := finished;
                break; }
             [] tc_V2G_EVCC_Ongoing_Timer.timeout{
                v_EVSEprocessing := finished;
                setverdict(fail,"EVSEProcessing = Finished was not sent in time.");
                break;
             }
          }
       }
       if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing and
          not v_receivedOngoing) {
          setverdict(inconc, "The SUT did not repeat the ChargeParameter" &
                             " message sequence.");
       }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }
    else {setverdict(inconc,"Requested energy transfer mode is not supported by the SECC.");}
    return getverdict;
  }

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(in verdicttype v_vct)
                                                        runs on SECC_Tester
                                                       return verdicttype {

  var EVSEProcessingType v_EVSEprocessing := ongoing;
  var V2G_Message v_responseMessage;
  var charstring v_message;
  var boolean isSignatureOk := true;
  var boolean v_receivedOngoing := true;
  var float v_processingDummy := 5.0;
  var integer v_cnt := 0;

  if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing) {
     v_receivedOngoing := false;
  }

  if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeDC,
                                        vc_supportedEnergyTransferMode)) {

      while (v_EVSEprocessing == ongoing){
          log("-------------------------------------------------------------");
           log("EV maximum charging limits:");
          log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
              vc_EVMaximumCurrentLimit)));
          log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
              vc_EVMaximumVoltageLimit)));
          log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
              vc_EVMaximumPowerLimit)));
          log("-------------------------------------------------------------");

          tc_V2G_EVCC_Msg_Timer.start(
          par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

          if(PICS_SECC_CMN_DepartureTime) {

             if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                 pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                omit, vc_requestedEnergyTransferModeDC,
                                                vc_DepartureTime, vc_DC_EVStatus,
                                                vc_EVMaximumCurrentLimit,
                                                vc_EVMaximumPowerLimit,
                                                vc_EVMaximumVoltageLimit))));
             } else {
                 pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                par_CMN_MaxEntriesSAScheduleTuple,
                                                vc_requestedEnergyTransferModeDC,
                                                vc_DepartureTime, vc_DC_EVStatus,
                                                vc_EVMaximumCurrentLimit,
                                                vc_EVMaximumPowerLimit,
                                                vc_EVMaximumVoltageLimit))));
             }
          }
          else{

             if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                 pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                omit, vc_requestedEnergyTransferModeDC, omit,
                                                vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                                vc_EVMaximumPowerLimit,
                                                vc_EVMaximumVoltageLimit))));
             } else {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                        par_CMN_MaxEntriesSAScheduleTuple,
                                        vc_requestedEnergyTransferModeDC, omit,
                                        vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                        vc_EVMaximumPowerLimit,
                                        vc_EVMaximumVoltageLimit))));
        }
    }

    alt {
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                           md_CMN_MessageHeaderType_002(vc_SessionID),
                                           md_CMN_BodyType_001(
                                           mdw_SECC_DC_ChargeParameterDiscoveryRes_001(
                                           finished, none_, ?, ?))))
                                           -> value v_responseMessage {

            v_EVSEprocessing := finished;
            tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
            v_cnt := v_cnt + 1;
            vc_SASchedule_List := v_responseMessage.body.bodyElement.
                                  chargeParameterDiscoveryRes.sASchedules.
                                  sAScheduleList;
            vc_SAScheduleTupleId := vc_SASchedule_List.sAScheduleTuple_list[0].
                                    sAScheduleTupleID;
            vc_PMaxSchedule := v_responseMessage.body.bodyElement.
                               chargeParameterDiscoveryRes.sASchedules.sAScheduleList.
                               sAScheduleTuple_list[0].pMaxSchedule;
            vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
                                          chargeParameterDiscoveryRes.
                                          eVSEChargeParameter.
                                          dC_EVSEChargeParameter.eVSEMaximumCurrentLimit;
            f_check_PhysicalValue_valueRange(cc_eVSEMaximumCurrentLimit,
                                             vc_EVSEMaximumCurrentLimit, a,
                                             cc_maximum_value_physicalValue_case1,
                                             v_vct);
            vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
                                          chargeParameterDiscoveryRes.
                                          eVSEChargeParameter.
                                          dC_EVSEChargeParameter.eVSEMaximumVoltageLimit;
            f_check_PhysicalValue_valueRange(cc_eVSEMaximumVoltageLimit,
                                             vc_EVSEMaximumVoltageLimit, v,
                                             cc_maximum_value_physicalValue_case2,
                                             v_vct);
            vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
                                        chargeParameterDiscoveryRes.
                                        eVSEChargeParameter.
                                        dC_EVSEChargeParameter.eVSEMaximumPowerLimit;
            f_check_PhysicalValue_valueRange(cc_eVSEMaximumPowerLimit,
                                             vc_EVSEMaximumPowerLimit, w,
                                             cc_maximum_value_physicalValue_case4,
                                             v_vct);
            vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
                                          chargeParameterDiscoveryRes.
                                          eVSEChargeParameter.
                                          dC_EVSEChargeParameter.eVSEMinimumVoltageLimit;
            f_check_PhysicalValue_valueRange(cc_eVSEMinimumVoltageLimit,
                                             vc_EVSEMinimumVoltageLimit, v,
                                             cc_maximum_value_physicalValue_case2,
                                             v_vct);
            vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.
                                          chargeParameterDiscoveryRes.
                                          eVSEChargeParameter.
                                          dC_EVSEChargeParameter.eVSEMinimumCurrentLimit;
            f_check_PhysicalValue_valueRange(cc_eVSEMinimumCurrentLimit,
                                             vc_EVSEMinimumCurrentLimit, a,
                                             cc_maximum_value_physicalValue_case1,
                                             v_vct);
            vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
                                        chargeParameterDiscoveryRes.eVSEChargeParameter.
                                        dC_EVSEChargeParameter.eVSEPeakCurrentRipple;
            f_check_PhysicalValue_valueRange(cc_eVSEPeakCurrentRipple,
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                vc_EVSEPeakCurrentRipple, a,
                                cc_maximum_value_physicalValue_case1,
                                v_vct);
tc_V2G_EVCC_Msg_Timer.stop;
tc_V2G_EVCC_Ongoing_Timer.stop;
log("-------------------------------------------------------------");
log("EVSE maximum charging limits:");
log("EVSEMaximumCurrentLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumCurrentLimit)));
log("EVSEMaximumVoltageLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumVoltageLimit)));
log("EVSEMaximumPowerLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumPowerLimit)));
log("EVSEMinimumCurrentLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMinimumCurrentLimit)));
log("EVSEMinimumVoltageLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMinimumVoltageLimit)));
log("-------------------------------------------------------------");
    f_checkPMaxSchedulesStructure(vc_SASchedule_List,
                                  cc_maximum_value_physicalValue_case4,
                                  PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
                                  v_vct);
    if(getverdict == pass) {
        f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedule_List,
                                          v_vct);
    }
    if(getverdict == pass) {
        f_checkSalesTariffStructure(vc_SASchedule_List,
        PICS_SECC_CMN_MaxEntriesSAScheduleTuple, v_vct);
    }
    if(getverdict == pass) {
       f_checkSalesTariffTimeIntervals(vc_DepartureTime, vc_SASchedule_List,
                                          v_vct);
    }
if(PICS_CMN_CMN_IdentificationMode == pnC) {
   var idDigestValueMap resultIdDigestValueMap :=
   fx_getIdDigestMapFromScheduleListType(vc_SASchedule_List,
                                    par_V2G_EVCC_Sequence_Performance_Time);
   if(lengthof(resultIdDigestValueMap) > 0) {
       var Signature authSignature := v_responseMessage.header.signature_;
       var BodyType chargeParameterResBodyType := v_responseMessage.body;

       f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                   cc_algorithm_method_2,
                                   cc_algorithm_method_3,
                                   v_vct);

       if(getverdict == pass) {

               isSignatureOk := f_verifySignature(
                                chargeParameterResBodyType,
                                authSignature,
                                vc_contractSignatureCertChain.
                                subCertificates.
                                x509Certificate_list[0], v_vct);

           if(isSignatureOk) {
               setverdict(pass, "Valid signature of the sales tariff " &
                                   "was received.");
           }
           else {
               setverdict(v_vct, "Invalid signature of the sales " &
                                    "tariff was received.");
           }
       }
   } else {
       setverdict(inconc, "Sales Tariff signature verification " &
                          "could not be executed within defined " &
                          "performance time.");
   }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }
    if(getverdict == pass) {
        setverdict(pass,"ChargeParameterDiscoveryRes message " &
                        "was correct.");
    }
    vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
    tc_CMN_ProcessingTime_Timer.stop;
}
[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_002(vc_SessionID),
                                md_CMN_BodyType_001(
                                mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
                                none_, ?)))) -> value v_responseMessage {
  tc_V2G_EVCC_Msg_Timer.stop;
  v_receivedOngoing := true;
  v_cnt := v_cnt + 1;
  if(ispresent(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
                eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
                eVSEIsolationStatus)) {
      if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
         eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
         eVSEIsolationStatus != invalid) {
         logWarn("The field 'eVSEIsolationStatus' should be " &
                  "set to 'invalid'.");
      }
  }
  if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
       eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
       eVSEStatusCode != eVSE_Ready) {
     logWarn("The field 'eVSEStatusCode' should be set " &
             "to 'eVSE_Ready'.");
  }
  if(tc_V2G_EVCC_Ongoing_Timer.running==false) {
       tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
  }
  vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
                                chargeParameterDiscoveryRes.
                                eVSEChargeParameter.
                                dC_EVSEChargeParameter.eVSEMaximumCurrentLimit;
  f_check_PhysicalValue_valueRange(cc_eVSEMaximumCurrentLimit,
                                   vc_EVSEMaximumCurrentLimit, a,
                                cc_maximum_value_physicalValue_case1,
                                v_vct);
  vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
                                chargeParameterDiscoveryRes.
                                eVSEChargeParameter.
                                dC_EVSEChargeParameter.eVSEMaximumVoltageLimit;
  f_check_PhysicalValue_valueRange(cc_eVSEMaximumVoltageLimit,
                                   vc_EVSEMaximumVoltageLimit, v,
                                cc_maximum_value_physicalValue_case2,
                                v_vct);
  vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
                              chargeParameterDiscoveryRes.
                              eVSEChargeParameter.
                              dC_EVSEChargeParameter.eVSEMaximumPowerLimit;
  f_check_PhysicalValue_valueRange(cc_eVSEMaximumPowerLimit,
                                   vc_EVSEMaximumPowerLimit, w,
                                cc_maximum_value_physicalValue_case4,
                                v_vct);
  vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
                                chargeParameterDiscoveryRes.
                                eVSEChargeParameter.
                                dC_EVSEChargeParameter.eVSEMinimumVoltageLimit;
  f_check_PhysicalValue_valueRange(cc_eVSEMinimumVoltageLimit,
                                   vc_EVSEMinimumVoltageLimit, v,
                                cc_maximum_value_physicalValue_case2,
                                v_vct);
  vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.
                                chargeParameterDiscoveryRes.
                                eVSEChargeParameter.
                                dC_EVSEChargeParameter.eVSEMinimumCurrentLimit;
  f_check_PhysicalValue_valueRange(cc_eVSEMinimumCurrentLimit,
                                   vc_EVSEMinimumCurrentLimit, a,
                                cc_maximum_value_physicalValue_case1,
                                v_vct);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
                                        chargeParameterDiscoveryRes.eVSEChargeParameter.
                                        dC_EVSEChargeParameter.eVSEPeakCurrentRipple;
            f_check_PhysicalValue_valueRange(cc_eVSEPeakCurrentRipple,
                                             vc_EVSEPeakCurrentRipple, a,
                                          cc_maximum_value_physicalValue_case1,
                                          v_vct);
            log("------------------------------------------------------------");
            log("EVSE maximum charging limits:");
            log("EVSEMaximumCurrentLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
                 vc_EVSEMaximumCurrentLimit)));
            log("EVSEMaximumVoltageLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
                 vc_EVSEMaximumVoltageLimit)));
            log("EVSEMaximumPowerLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
                 vc_EVSEMaximumPowerLimit)));
            log("EVSEMinimumCurrentLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
                 vc_EVSEMinimumCurrentLimit)));
            log("EVSEMinimumVoltageLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
                 vc_EVSEMinimumVoltageLimit)));
            log("------------------------------------------------------------");

            setverdict(pass,"ChargeParameterDiscoveryRes message (Ongoing) " &
                            "was correct.");
         }
         [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                        "was misleadingly " &
                                                        "terminated by the SUT.") {
            v_EVSEprocessing := finished;
            break; }
         [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                         {chargeParameterDiscoveryRes := ?}) {
            v_EVSEprocessing := finished;
            break;
         }
         [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                      "but cannot be decoded.") {
            v_EVSEprocessing := finished;
            break;
         }
         [] a_SECC_Unexpected_Message_Content(v_vct, "Received ChargeParameterDiscoveryRes " &
                                                     "message, but with invalid " &
                                                     "data content.") {
            v_EVSEprocessing := finished;
            break;
         }
         [] a_SECC_Timeout(v_vct) {
            v_EVSEprocessing := finished;
            break; }
         [] tc_V2G_EVCC_Ongoing_Timer.timeout{
            v_EVSEprocessing := finished;
            setverdict(v_vct,"EVSEProcessing = Finished was not sent in time.");
            break;
         }
      }
   }
   if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing and
      not v_receivedOngoing) {
      setverdict(inconc, "The SUT did not repeat the ChargeParameter" &
                         " message sequence.");
   }
}
else {setverdict(inconc,"Requested energy transfer mode is not supported by the SECC.");}
return getverdict;
}

function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_007() runs on SECC_Tester
                                                         return verdicttype {

   var EVSEProcessingType v_EVSEprocessing := ongoing;
   var V2G_Message v_responseMessage;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var charstring v_message;
var boolean isSignatureOk := true;
var boolean v_receivedOngoing := true;

if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing) {
   v_receivedOngoing := false;
}

if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeDC,
                                      vc_supportedEnergyTransferMode)) {

    while (v_EVSEprocessing == ongoing){
         log("-------------------------------------------------------------");
            log("EV maximum charging limits:");
         log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
             vc_EVMaximumCurrentLimit)));
         log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
             vc_EVMaximumVoltageLimit)));
         log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
             vc_EVMaximumPowerLimit)));
         log("-------------------------------------------------------------");

         tc_V2G_EVCC_Msg_Timer.start(
         par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

         if(PICS_SECC_CMN_DepartureTime) {

            if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                omit, vc_requestedEnergyTransferModeDC,
                                                vc_DepartureTime, vc_DC_EVStatus,
                                                vc_EVMaximumCurrentLimit,
                                                vc_EVMaximumPowerLimit,
                                                vc_EVMaximumVoltageLimit))));
            } else {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                par_CMN_MaxEntriesSAScheduleTuple,
                                                vc_requestedEnergyTransferModeDC,
                                                vc_DepartureTime, vc_DC_EVStatus,
                                                vc_EVMaximumCurrentLimit,
                                                vc_EVMaximumPowerLimit,
                                                vc_EVMaximumVoltageLimit))));
            }
         }
         else{

            if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                omit, vc_requestedEnergyTransferModeDC, omit,
                                                vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                                vc_EVMaximumPowerLimit,
                                                vc_EVMaximumVoltageLimit))));
            } else {
                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                par_CMN_MaxEntriesSAScheduleTuple,
                                                vc_requestedEnergyTransferModeDC, omit,
                                                vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                                vc_EVMaximumPowerLimit,
                                                vc_EVMaximumVoltageLimit))));
            }
         }

        alt {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_002(vc_SessionID),
                                md_CMN_BodyType_001(
                                mdw_SECC_DC_ChargeParameterDiscoveryRes_001(
                                finished, none_, ?, ?))))
                                -> value v_responseMessage {

        v_EVSEprocessing := finished;
        vc_SASchedule_List := v_responseMessage.body.bodyElement.
                              chargeParameterDiscoveryRes.sASchedules.
                              sAScheduleList;
        vc_PMaxSchedule := v_responseMessage.body.bodyElement.
                           chargeParameterDiscoveryRes.sASchedules.
                           sAScheduleList.
                           sAScheduleTuple_list[0].pMaxSchedule;
        vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.
                                      eVSEChargeParameter.
                                      dC_EVSEChargeParameter.
                                      eVSEMaximumCurrentLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMaximumCurrentLimit,
                                         vc_EVSEMaximumCurrentLimit, a,
                                      cc_maximum_value_physicalValue_case1,
                                      fail);
        vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.
                                      eVSEChargeParameter.
                                      dC_EVSEChargeParameter.
                                      eVSEMaximumVoltageLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMaximumVoltageLimit,
                                         vc_EVSEMaximumVoltageLimit, v,
                                      cc_maximum_value_physicalValue_case2,
                                      fail);
        vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
                                    chargeParameterDiscoveryRes.
                                    eVSEChargeParameter.
                                    dC_EVSEChargeParameter.eVSEMaximumPowerLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMaximumPowerLimit,
                                         vc_EVSEMaximumPowerLimit, w,
                                      cc_maximum_value_physicalValue_case4,
                                      fail);
        vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.
                                      eVSEChargeParameter.
                                      dC_EVSEChargeParameter.
                                      eVSEMinimumVoltageLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMinimumVoltageLimit,
                                         vc_EVSEMinimumVoltageLimit, v,
                                      cc_maximum_value_physicalValue_case2,
                                      fail);
        vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.
                                      eVSEChargeParameter.
                                      dC_EVSEChargeParameter.
                                      eVSEMinimumCurrentLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMinimumCurrentLimit,
                                         vc_EVSEMinimumCurrentLimit, a,
                                      cc_maximum_value_physicalValue_case1,
                                      fail);
        vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
                                    chargeParameterDiscoveryRes.
                                    eVSEChargeParameter.
                                    dC_EVSEChargeParameter.
                                    eVSEPeakCurrentRipple;
        f_check_PhysicalValue_valueRange(cc_eVSEPeakCurrentRipple,
                                         vc_EVSEPeakCurrentRipple, a,
                                      cc_maximum_value_physicalValue_case1,
                                      fail);
        tc_V2G_EVCC_Msg_Timer.stop;
        tc_V2G_EVCC_Ongoing_Timer.stop;
        log("-------------------------------------------------------------");
        log("EVSE maximum charging limits:");
        log("EVSEMaximumCurrentLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumCurrentLimit)));
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
log("EVSEMaximumVoltageLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumVoltageLimit)));
log("EVSEMaximumPowerLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumPowerLimit)));
log("EVSEMinimumCurrentLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMinimumCurrentLimit)));
log("EVSEMinimumVoltageLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVSEMinimumVoltageLimit)));
log("-------------------------------------------------------------");
f_checkPMaxSchedulesStructure(vc_SASchedule_List,
                              cc_maximum_value_physicalValue_case4,
                              PICS_SECC_CMN_MaxEntriesSAScheduleTuple,
                              fail);
if(getverdict == pass) {
    f_checkPMaxSchedulesTimeIntervals(vc_DepartureTime, vc_SASchedule_List,
                                      fail);
}
if(getverdict == pass) {
    f_checkSalesTariffStructure(vc_SASchedule_List,
    PICS_SECC_CMN_MaxEntriesSAScheduleTuple, fail);
}
if(getverdict == pass) {
   f_checkSalesTariffTimeIntervals(vc_DepartureTime, vc_SASchedule_List,
                                   fail);
}
if(PICS_CMN_CMN_IdentificationMode == pnC) {
   var idDigestValueMap resultIdDigestValueMap :=
   fx_getIdDigestMapFromScheduleListType(vc_SASchedule_List,
                                         par_V2G_EVCC_Sequence_Performance_Time);
   if(lengthof(resultIdDigestValueMap) > 0) {
       var Signature authSignature := v_responseMessage.header.signature_;
       var BodyType chargeParameterResBodyType := v_responseMessage.body;

       f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                   cc_algorithm_method_2,
                                   cc_algorithm_method_3,
                                   fail);

       if(getverdict == pass) {

               isSignatureOk := f_verifySignature(
                                chargeParameterResBodyType,
                                authSignature,
                                vc_contractSignatureCertChain.
                                subCertificates.
                                x509Certificate_list[0], fail);

           if(isSignatureOk) {
               setverdict(pass, "Valid signature of the sales tariff " &
                                "was received.");
           }
           else {
               setverdict(fail, "Invalid signature of the sales tariff " &
                                "was received.");
           }
       }
   } else {
      setverdict(inconc, "Sales Tariff signature verification " &
                         "could not be executed within defined " &
                         "performance time.");
   }
}
if(getverdict == pass) {
    if(f_compareToPreviousSAScheduleTuple(vc_SASchedule_List,
                                          vc_SAScheduleTupleId)) {
        setverdict(pass,"ChargeParameterDiscoveryRes message " &
                        "was correct. A valid SAScheduleTuple " &
                        "was received after paused session.");
    }
    else {
        setverdict(fail,"Invalid SAScheduleTuple was received after " &
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                "paused session.");
                }
            }
        }
    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_002(vc_SessionID),
                                md_CMN_BodyType_001(
                                mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
                                none_, ?)))) -> value v_responseMessage {
        tc_V2G_EVCC_Msg_Timer.stop;
        if(ispresent(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
                    eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
                    eVSEIsolationStatus)) {
            if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
               eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
               eVSEIsolationStatus != invalid) {
               logWarn("The field 'eVSEIsolationStatus' should be " &
                       "set to 'invalid'.");
            }
        }
        if(v_responseMessage.body.bodyElement.chargeParameterDiscoveryRes.
           eVSEChargeParameter.dC_EVSEChargeParameter.dC_EVSEStatus.
           eVSEStatusCode != eVSE_Ready) {
           logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
        }
        if(tc_V2G_EVCC_Ongoing_Timer.running==false){
             tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
        }
        vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.
                                      eVSEChargeParameter.
                                      dC_EVSEChargeParameter.
                                      eVSEMaximumCurrentLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMaximumCurrentLimit,
                                         vc_EVSEMaximumCurrentLimit, a,
                                         cc_maximum_value_physicalValue_case1,
                                         fail);
        vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.
                                      eVSEChargeParameter.
                                      dC_EVSEChargeParameter.
                                      eVSEMaximumVoltageLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMaximumVoltageLimit,
                                         vc_EVSEMaximumVoltageLimit, v,
                                         cc_maximum_value_physicalValue_case2,
                                         fail);
        vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
                                    chargeParameterDiscoveryRes.
                                    eVSEChargeParameter.
                                    dC_EVSEChargeParameter.
                                    eVSEMaximumPowerLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMaximumPowerLimit,
                                         vc_EVSEMaximumPowerLimit, w,
                                         cc_maximum_value_physicalValue_case4,
                                         fail);
        vc_EVSEMinimumVoltageLimit := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.
                                      eVSEChargeParameter.
                                      dC_EVSEChargeParameter.
                                      eVSEMinimumVoltageLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMinimumVoltageLimit,
                                         vc_EVSEMinimumVoltageLimit, v,
                                         cc_maximum_value_physicalValue_case2,
                                         fail);
        vc_EVSEMinimumCurrentLimit := v_responseMessage.body.bodyElement.
                                      chargeParameterDiscoveryRes.
                                      eVSEChargeParameter.
                                      dC_EVSEChargeParameter.
                                      eVSEMinimumCurrentLimit;
        f_check_PhysicalValue_valueRange(cc_eVSEMinimumCurrentLimit,
                                         vc_EVSEMinimumCurrentLimit, a,
                                         cc_maximum_value_physicalValue_case1,
                                         fail);
        vc_EVSEPeakCurrentRipple := v_responseMessage.body.bodyElement.
                                    chargeParameterDiscoveryRes.eVSEChargeParameter.
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                dC_EVSEChargeParameter.eVSEPeakCurrentRipple;
                        f_check_PhysicalValue_valueRange(cc_eVSEPeakCurrentRipple,
                                                         vc_EVSEPeakCurrentRipple, a,
                                                         cc_maximum_value_physicalValue_case1,
                                                         fail);
                        log("------------------------------------------------------------");
                        log("EVSE maximum charging limits:");
                        log("EVSEMaximumCurrentLimit: " & toString<float>(
                        f_CMN_PhysicalValue_GetValue(
                             vc_EVSEMaximumCurrentLimit)));
                        log("EVSEMaximumVoltageLimit: " & toString<float>(
                        f_CMN_PhysicalValue_GetValue(
                             vc_EVSEMaximumVoltageLimit)));
                        log("EVSEMaximumPowerLimit: " & toString<float>(
                        f_CMN_PhysicalValue_GetValue(
                             vc_EVSEMaximumPowerLimit)));
                        log("EVSEMinimumCurrentLimit: " & toString<float>(
                        f_CMN_PhysicalValue_GetValue(
                             vc_EVSEMinimumCurrentLimit)));
                        log("EVSEMinimumVoltageLimit: " & toString<float>(
                        f_CMN_PhysicalValue_GetValue(
                             vc_EVSEMinimumVoltageLimit)));
                        log("------------------------------------------------------------");

                        setverdict(pass,"ChargeParameterDiscoveryRes message (Ongoing) " &
                                        "was correct.");
                    }
                    [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                                  "was misleadingly " &
                                                                 "terminated by the SUT.") {
                       v_EVSEprocessing := finished;
                       break; }
                    [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                                     {chargeParameterDiscoveryRes := ?}) {
                       v_EVSEprocessing := finished;
                       break;
                    }
                    [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                                "but cannot be decoded.") {
                       v_EVSEprocessing := finished;
                       break;
                    }
                    [] a_SECC_Unexpected_Message_Content(fail, "Received " &
                                                               "ChargeParameterDiscoveryRes " &
                                                               "message, but with invalid " &
                                                               "data content.") {
                       v_EVSEprocessing := finished;
                       break;
                    }
                    [] a_SECC_Timeout(fail) {
                       v_EVSEprocessing := finished;
                       break; }
                    [] tc_V2G_EVCC_Ongoing_Timer.timeout{
                       v_EVSEprocessing := finished;
                       setverdict(fail,"EVSEProcessing = Finished was not sent in time.");
                       break;
                    }
                 }
              }
              if(PIXIT_SECC_CMN_ChargeParameterDiscoveryOngoing == ongoing and
                 not v_receivedOngoing) {
                 setverdict(inconc, "The SUT did not repeat the ChargeParameter" &
                                    " message sequence.");
              }
           }
           else {setverdict(inconc,"Requested energy transfer mode is not supported by the SECC.");}
           return getverdict;
        }

        function f_SECC_DC_TB_VTB_ChargeParameterDiscovery_008() runs on SECC_Tester
                                                                 return verdicttype  {

           var EVSEProcessingType v_EVSEprocessing := ongoing;
           var charstring v_message;
           var boolean isShutdownOSC := false;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var boolean isClosedTCP := false;
var boolean isErrorMessage := false;
var float v_timeShift := 1.0;
var charstring v_charstring := "";

if(f_checkRequestedEnergyTransfermode(vc_requestedEnergyTransferModeDC,
                                      vc_supportedEnergyTransferMode)) {

    while (v_EVSEprocessing == ongoing){
        log("------------------------------------------------------------");
           log("EV maximum charging limits:");
        log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumCurrentLimit)));
        log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumVoltageLimit)));
        log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumPowerLimit)));
        log("------------------------------------------------------------");

        tc_V2G_EVCC_Msg_Timer.start(
        par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

        if(PICS_SECC_CMN_DepartureTime) {

           if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
               pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                          omit, vc_requestedEnergyTransferModeDC,
                                          vc_DepartureTime, vc_DC_EVStatus,
                                          vc_EVMaximumCurrentLimit,
                                          vc_EVMaximumPowerLimit,
                                          vc_EVMaximumVoltageLimit))));
           } else {
               pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                          par_CMN_MaxEntriesSAScheduleTuple,
                                          vc_requestedEnergyTransferModeDC,
                                          vc_DepartureTime, vc_DC_EVStatus,
                                          vc_EVMaximumCurrentLimit,
                                          vc_EVMaximumPowerLimit,
                                          vc_EVMaximumVoltageLimit))));
           }
        }
        else{

           if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
               pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                          omit, vc_requestedEnergyTransferModeDC, omit,
                                          vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                          vc_EVMaximumPowerLimit,
                                          vc_EVMaximumVoltageLimit))));
           } else {
               pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                          par_CMN_MaxEntriesSAScheduleTuple,
                                          vc_requestedEnergyTransferModeDC, omit,
                                          vc_DC_EVStatus, vc_EVMaximumCurrentLimit,
                                          vc_EVMaximumPowerLimit,
                                          vc_EVMaximumVoltageLimit))));
           }
        }

        alt {
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_002(vc_SessionID),
                                            md_CMN_BodyType_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                mdw_SECC_DC_ChargeParameterDiscoveryRes_001(
                                                finished, none_, ?, *)))) {

                    v_EVSEprocessing := finished;
                    setverdict(inconc,"EVSEProcessingType 'Finished' was not expected. " &
                                      "Please check PIXIT configuration.");
                }
                []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_002(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
                                                 none_, ?)))) {

                  if(tc_V2G_EVCC_Ongoing_Timer.running==false)
                   {
                       tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout - v_timeShift);
                   }
                }
                [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                              "was misleadingly " &
                                                            "terminated by the SUT.") {
                   v_EVSEprocessing := finished;
                   break;
                }
                [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {chargeParameterDiscoveryRes := ?}) {
                   v_EVSEprocessing := finished;
                   break;
                }
                [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                            "but cannot be decoded.") {
                   v_EVSEprocessing := finished;
                   break;
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                                           "message, but with invalid " &
                                                           "data content.") {
                   v_EVSEprocessing := finished;
                   break;
                }
                [] a_SECC_Timeout(fail) {
                   v_EVSEprocessing := finished;
                   break;
                }
                [] tc_V2G_EVCC_Ongoing_Timer.timeout{

                      f_SECC_setIsConfirmationFlagDC(true);
                      f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                                       vc_FrequencyOff);
                      f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                                       vc_DutyCycleOscOff);
                      deactivate(vc_Default_IEC_61851_ListenerBehavior);

                  tc_V2G_SECC_Ongoing_Timer.start(v_timeShift -
                                                  par_CMN_Transmission_Delay);

                  alt {
                      []
                      a_Shutdown_Oscillator(pass){
                            isShutdownOSC := true;
                            if(isClosedTCP == false) {repeat;}
                      }
                      [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(

md_CMN_MessageHeaderType_002(vc_SessionID),
                                                          md_CMN_BodyType_001(

mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
                                                          fAILED)))) {

                            setverdict(pass,"ChargeParameterDiscoveryRes with responseCode " &
                                            "'fAILED' is successful.");

                            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                                         e_getPortStatus));
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                isErrorMessage := true;
                                repeat;
                            }
                            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                                          "terminated by the SUT.") {
                                isClosedTCP := true;
                                tc_CMN_TCP_Connection_Termination_Timer.stop;
                                if(isShutdownOSC == false) {
                                   logWarn("SECC oscillator was not shut down before the " &
                                           "TCP connection was terminated.");}
                                pt_HAL_61851_Internal_Port.clear;
                            }
                            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(

md_CMN_MessageHeaderType_002(vc_SessionID),
                                                                  md_CMN_BodyType_001(

mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
                                                                  none_, ?)))) {

                               tc_V2G_EVCC_Msg_Timer.stop;

tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq);

                               if(PICS_SECC_CMN_DepartureTime) {

                                 if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                                     pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(

md_CMN_MessageHeaderType_001(vc_SessionID),
                                                                   md_CMN_BodyType_001(

md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                                   omit,
vc_requestedEnergyTransferModeDC,
                                                                   vc_DepartureTime, vc_DC_EVStatus,
                                                                   vc_EVMaximumCurrentLimit,
                                                                   vc_EVMaximumPowerLimit,
                                                                   vc_EVMaximumVoltageLimit))));
                                 } else {
                                     pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(

md_CMN_MessageHeaderType_001(vc_SessionID),
                                                                   md_CMN_BodyType_001(

md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                                   par_CMN_MaxEntriesSAScheduleTuple,
                                                                   vc_requestedEnergyTransferModeDC,
                                                                   vc_DepartureTime, vc_DC_EVStatus,
                                                                   vc_EVMaximumCurrentLimit,
                                                                   vc_EVMaximumPowerLimit,
                                                                   vc_EVMaximumVoltageLimit))));
                                 }
                               }
                               else{

                                 if(not PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                                     pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(

md_CMN_MessageHeaderType_001(vc_SessionID),
                                                                   md_CMN_BodyType_001(

md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                                   omit,
vc_requestedEnergyTransferModeDC, omit,
                                                                   vc_DC_EVStatus,
vc_EVMaximumCurrentLimit,
                                                                   vc_EVMaximumPowerLimit,
                                                                   vc_EVMaximumVoltageLimit))));
                                 } else {
                                     pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(

md_CMN_MessageHeaderType_001(vc_SessionID),
                                                                   md_CMN_BodyType_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
md_SECC_DC_ChargeParameterDiscoveryReq_001(
                                                                    par_CMN_MaxEntriesSAScheduleTuple,
                                                                    vc_requestedEnergyTransferModeDC,
omit,
                                                                    vc_DC_EVStatus,
vc_EVMaximumCurrentLimit,
                                                                    vc_EVMaximumPowerLimit,
                                                                    vc_EVMaximumVoltageLimit))));
                                }
                            }
                            repeat;
                        }
                        [] tc_V2G_SECC_Ongoing_Timer.timeout {
                            tc_CMN_TCP_Connection_Termination_Timer.start(
                            par_CMN_TCP_Connection_Termination_Timeout);
                            repeat;
                        }
                        [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                                   "but TCP connection was " &
                                                                   ">>NOT<< terminated by " &
                                                                   "the SUT");
                        [] a_SECC_TCP_Connection_Termination_Timeout() {
                            v_charstring := v_charstring & "SUT has not terminated " &
                                    "the TCP connection " &
                                    "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
                        }
                    }
                    if(not isErrorMessage) {
                        v_charstring := v_charstring & "SUT has not sent a " &
                                        "ChargeParameterDiscoveryRes " &
                                        "message with responseCode 'fAILED' " &
                                        "or other message data content was not conform. ";
                    }
                    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
                       setverdict(fail,v_charstring);
                    }
                    v_EVSEprocessing := finished;
                    break;
                }
            }
        }
    }
    else {setverdict(inconc,"Requested energy transfer mode is not supported by the SECC.");}
    return getverdict;
}

function f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_001() runs on SECC_Tester
                                                          return verdicttype {

   var charstring v_message;
   var boolean isShutdownOSC := false;
   var boolean isClosedTCP := false;
   var boolean isErrorMessage := false;
   var charstring v_charstring := "";

          f_SECC_setIsConfirmationFlagDC(true);
       f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                        vc_FrequencyOff);
       f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                        vc_DutyCycleOscOff);

   tc_V2G_EVCC_Msg_Timer.start(
   par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
   tc_CMN_TCP_Connection_Termination_Timer.start(
   par_CMN_TCP_Connection_Termination_Timeout);

   pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

   alt {
      []
      a_Shutdown_Oscillator(pass){
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            isShutdownOSC := true;
            if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                          fAILED_SequenceError)))) {
             tc_V2G_EVCC_Msg_Timer.stop;
             setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                            "'fAILED_SequenceError' is successful.");

             pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
             isErrorMessage := true;
             repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                "terminated by the SUT.") {
            isClosedTCP := true;
            tc_CMN_TCP_Connection_Termination_Timer.stop;
            if(isShutdownOSC == false) {
               logWarn("SECC oscillator was not shut down before the " &
                       "TCP connection was terminated.");}
            pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                       {serviceDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received ChargeParameterDiscoveryRes " &
                                                 "message, but with invalid " &
                                                 "data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                        "message with responseCode 'fAILED_SequenceError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_ChargeParameterDiscovery_002() runs on SECC_Tester
                                                          return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var float v_timeShift := 1.0;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
                 v_timeShift);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                        "terminated by the SUT " &
                                        "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_checkRequestedEnergyTransfermode(EnergyTransferModeType
                                            v_requestedEnergyTransferMode,
                                            SupportedEnergyTransferModeType
                                            v_supportedEnergyTransferMode)
                                            return boolean {

    for (var integer i:=0; i<sizeof(v_supportedEnergyTransferMode.
                                    energyTransferMode_list); i:=i + 1) {
        if(v_supportedEnergyTransferMode.energyTransferMode_list[i] ==
           v_requestedEnergyTransferMode){
           return true;
        }
    }
    return false;
}

function f_checkPMaxSchedulesStructure(in SAScheduleListType
                                       vc_SASchedule_List, in float
                                       cc_maximum_value_physicalValue_case4,
                                       in boolean v_isMaxEntriesLimit,
                                       in verdicttype v_vct) {

    const charstring cc_pMax := "PMax";

    for (var integer i:=0; i<sizeof(vc_SASchedule_List.
                                    sAScheduleTuple_list); i:=i + 1) {
      if(vc_SASchedule_List.sAScheduleTuple_list[i].sAScheduleTupleID < 1 or
         vc_SASchedule_List.sAScheduleTuple_list[i].sAScheduleTupleID > 255) {
                setverdict(v_vct,"The SUT shall use the values 1 to 255 for " &
                                 "the parameter SAScheduleTupleID.");
                break;
      }
      for (var integer j:=i+1; j<sizeof(vc_SASchedule_List.
                                        sAScheduleTuple_list); j:=j + 1) {
          if(vc_SASchedule_List.sAScheduleTuple_list[i].sAScheduleTupleID ==
             vc_SASchedule_List.sAScheduleTuple_list[j].sAScheduleTupleID){
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    setverdict(v_vct,"The element 'SAScheduleTupleID' is not unique.");
                    break;
            }
        }
        var integer listLength := sizeof(vc_SASchedule_List.sAScheduleTuple_list[i].
                                        pMaxSchedule.pMaxScheduleEntry_list);

        if(listLength > par_CMN_MaxEntriesSAScheduleTuple and
           PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                setverdict(v_vct,"The SUT did not have considered 'MaxEntriesSAScheduleTuple' " &
                                 "within SAScheduleTuple (PMaxScheduleEntryList).");
                break;
        }
        for (var integer k:=0; k<listLength; k:=k + 1) {
             var PhysicalValueType v_pmax := vc_SASchedule_List.sAScheduleTuple_list[i].
                                             pMaxSchedule.pMaxScheduleEntry_list[k].pMax;
             f_check_PhysicalValue_valueRange(cc_pMax, v_pmax, w,
                                              cc_maximum_value_physicalValue_case4,
                                              v_vct);
        }
        for (var integer n:=0; n<listLength-1; n:=n + 1) {
            if(ispresent(vc_SASchedule_List.sAScheduleTuple_list[i].pMaxSchedule.
                         pMaxScheduleEntry_list[n].timeInterval.
                         relativeTimeInterval.duration)){
                    setverdict(v_vct,"The element 'duration' shall only be used for " &
                                     "the last interval of the PMaxSchedule.");
                    break;
            }
        }
        if(not(ispresent(vc_SASchedule_List.sAScheduleTuple_list[i].pMaxSchedule.
                         pMaxScheduleEntry_list[listLength-1].timeInterval.
                         relativeTimeInterval.duration))) {
                setverdict(v_vct,"The element 'duration' shall be used for the last " &
                                 "interval of the PMaxSchedule.");
        }
      }
    }

    function f_checkACPMaxScheduleEntries(in SAScheduleListType vc_SASchedule_List,
                                          in float v_pMax, in verdicttype v_vct) {
      for (var integer i:=0; i<sizeof(vc_SASchedule_List.
                                      sAScheduleTuple_list); i:=i + 1) {
        var integer listLength := sizeof(vc_SASchedule_List.sAScheduleTuple_list[i].
                                         pMaxSchedule.pMaxScheduleEntry_list);
        for (var integer m:=0; m<listLength; m:=m + 1) {
            if(v_pMax < f_CMN_PhysicalValue_GetValue(vc_SASchedule_List.
               sAScheduleTuple_list[i].pMaxSchedule.pMaxScheduleEntry_list[m].pMax)) {
                setverdict(v_vct,"Invalid PMaxSchedule entry. The SUT did not " &
                                 "calculated the PMax values based " &
                                 "on the parameter EVSENominalVoltage.");
            }
        }
      }
    }

    function f_checkPMaxSchedulesTimeIntervals(XSDAUX.unsignedInt vc_DepartureTime,
                                               SAScheduleListType vc_SASchedule_List,
                                               in verdicttype v_vct){

      var integer v_time24h := 86400;

      for (var integer i:=0; i<sizeof(vc_SASchedule_List); i:=i + 1) {
          var SAScheduleTupleType v_scheduleTuple := vc_SASchedule_List.
                                                     sAScheduleTuple_list[i];
          var integer v_sumPmaxSchedule := 0;

          var integer v_start := v_scheduleTuple.pMaxSchedule.pMaxScheduleEntry_list[0].
                                 timeInterval.relativeTimeInterval.start_;
          var integer v_stop := v_scheduleTuple.pMaxSchedule.pMaxScheduleEntry_list[
                                sizeof(v_scheduleTuple.pMaxSchedule.
                                pMaxScheduleEntry_list)-1].timeInterval.
                                relativeTimeInterval.start_;

          if(ispresent(v_scheduleTuple.pMaxSchedule.pMaxScheduleEntry_list[
                       sizeof(v_scheduleTuple.pMaxSchedule.
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                    pMaxScheduleEntry_list)-1].timeInterval.
                    relativeTimeInterval.duration)) {

            var integer v_duration := v_scheduleTuple.pMaxSchedule.pMaxScheduleEntry_list[
                                      sizeof(v_scheduleTuple.pMaxSchedule.
                                      pMaxScheduleEntry_list)-1].timeInterval.
                                      relativeTimeInterval.duration;
            v_sumPmaxSchedule := v_stop - v_start + v_duration;

            if(PICS_SECC_CMN_DepartureTime) {
                if(v_sumPmaxSchedule != vc_DepartureTime) {
                        setverdict(v_vct, "The sum of the individual time intervals " &
                                          "(PMaxSchedule) did not match " &
                                          "the period of time indicated by the " &
                                          "EVCC (DepartureTime).");
                }
            }
            else {
                if(v_sumPmaxSchedule < v_time24h) {
                        setverdict(v_vct, "The sum of the individual time " &
                                          "intervals (PMaxSchedule) was " &
                                          "smaller than 24h.");
                }
            }
        }
    }
}

function f_compareToPreviousSAScheduleTuple(in SAScheduleListType vc_SASchedule_List,
                                            SAIDType vc_SAScheduleTupleId)
                                            return boolean {

    for (var integer i:=0; i<sizeof(vc_SASchedule_List); i:=i + 1) {
        if(vc_SASchedule_List.sAScheduleTuple_list[0].sAScheduleTupleID ==
           vc_SAScheduleTupleId) {
           return true;
        }
    }
    return false;
}

function f_checkSalesTariffStructure(in SAScheduleListType
                                     vc_SASchedule_List,
                                     in boolean v_isMaxEntriesLimit,
                                     in verdicttype v_vct) {

    for (var integer i:=0; i<sizeof(vc_SASchedule_List.
                                    sAScheduleTuple_list); i:=i + 1) {

      var integer listLength := sizeof(vc_SASchedule_List.sAScheduleTuple_list[i].salesTariff.
                                       salesTariffEntry_list);

      if(listLength > par_CMN_MaxEntriesSAScheduleTuple and
         PICS_SECC_CMN_MaxEntriesSAScheduleTuple) {
                setverdict(v_vct,"The SUT did not have considered 'MaxEntriesSAScheduleTuple' " &
                                 "within SAScheduleTuple (SalesTariffEntryList).");
                break;
      }
    }
}

function f_checkSalesTariffTimeIntervals(XSDAUX.unsignedInt vc_DepartureTime,
                                         SAScheduleListType vc_SASchedule_List,
                                         in verdicttype v_vct){

    var integer v_time24h := 86400;

    for (var integer i:=0; i<sizeof(vc_SASchedule_List); i:=i + 1) {
        var SAScheduleTupleType v_scheduleTuple := vc_SASchedule_List.
                                                   sAScheduleTuple_list[i];
        var integer v_sumSalesTariff := 0;

        var integer v_start := v_scheduleTuple.salesTariff.salesTariffEntry_list[0].
                               timeInterval.relativeTimeInterval.start_;
        var integer v_stop := v_scheduleTuple.salesTariff.salesTariffEntry_list[
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                sizeof(v_scheduleTuple.salesTariff.
                                salesTariffEntry_list)-1].timeInterval.
                                relativeTimeInterval.start_;

            if(ispresent(v_scheduleTuple.salesTariff.salesTariffEntry_list[
                         sizeof(v_scheduleTuple.salesTariff.
                         salesTariffEntry_list)-1].timeInterval.
                         relativeTimeInterval.duration)) {

                var integer v_duration := v_scheduleTuple.salesTariff.salesTariffEntry_list[
                                          sizeof(v_scheduleTuple.salesTariff.
                                          salesTariffEntry_list)-1].timeInterval.
                                          relativeTimeInterval.duration;
                v_sumSalesTariff := v_stop - v_start + v_duration;

                if(PICS_SECC_CMN_DepartureTime) {
                    if(v_sumSalesTariff != vc_DepartureTime) {
                            setverdict(v_vct, "The sum of the individual time intervals " &
                                              "(SalesTariff) did not match " &
                                              "the period of time indicated by the " &
                                              "EVCC (DepartureTime).");
                    }
                }
                else {
                    if(v_sumSalesTariff < v_time24h) {
                            setverdict(v_vct, "The sum of the individual time " &
                                              "intervals (SalesTariff) was " &
                                              "smaller than 24h.");

                    }
                }
            }
        }
    }

    function f_checkEVSEChargeParameter(in PhysicalValueType vc_EVSEMaximumCurrentLimit,
                                        in PhysicalValueType vc_EVSEMaximumVoltageLimit,
                                        in PhysicalValueType vc_EVMaximumCurrentLimit,
                                        in PhysicalValueType vc_EVMaximumVoltageLimit,
                                        in PhysicalValueType vc_EVTargetVoltage,
                                        in verdicttype v_vct) {

         if(f_CMN_PhysicalValue_GetValue(vc_EVMaximumCurrentLimit) >
             f_CMN_PhysicalValue_GetValue(vc_EVSEMaximumCurrentLimit)) {
                 logWarn("The EV maximum limits are higher than " &
                         "the EVSE maximum limits. " &
                         "The EV adapts the corresponding parameter " &
                         "to the EVSE limits.");
                 vc_EVMaximumCurrentLimit := vc_EVSEMaximumCurrentLimit;
         }
         if(f_CMN_PhysicalValue_GetValue(vc_EVMaximumVoltageLimit) >
            f_CMN_PhysicalValue_GetValue(vc_EVSEMaximumVoltageLimit)) {

             logWarn("The EV maximum limits are higher than " &
                     "the EVSE maximum limits. " &
                     "The EV adapts the corresponding parameter " &
                     "to the EVSE limits.");
             vc_EVMaximumVoltageLimit := vc_EVSEMaximumVoltageLimit;
         }
    }
}
```

I.2.2.8 SECC functions for PowerDelivery

```
module TestBehavior_SECC_PowerDelivery {

    import from ComponentsAndPorts all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_CMN_ServiceDiscovery all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from TestBehavior_SECC_CommonBehavior all;
import from Timer_15118_2 all;
import from Templates_SECC_ServiceDiscovery all;
import from DataStructure_HAL_61851 all;
import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
import from TTlibrary_Logging all;
import from Pics_15118 all;
import from Pics_15118_2 all;
import from Services_HAL_61851_1 all;
import from Templates_CMN_V2GTcpTlsPortControl all;

import from Pixit_15118 all;
import from Timer_15118 all;
import from LibFunctions_15118_2 all;

private function f_SECC_61851_State_CorDorB(in ChargeProgressType cpt,
                                            in HAL_61851_Listener listener,
                                            in boolean v_deactivate)
                                            runs on SECC_Tester {

    var IEC_61851_States v_state;
    // set 61851 state C or B depending on charge progress type
    if(cpt == start_) {
        v_state := PICS_SECC_CMN_EvStateInChargeProgress;
    } else {
        v_state := e_StateB;
    }

        if () {
            f_SECC_changeValidStateCondition(v_state);
            if(v_deactivate) {
                deactivate(vc_Default_IEC_61851_ListenerBehavior);
            }
            f_SECC_setState(v_state, listener);
        }
}

altstep a_AC_PowerDeliveryRes_Message(ChargeProgressType c, ResponseCodeType rc)
                                      runs on SECC_Tester {

  var V2G_Message v_responseMessage;
  [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_CMN_PowerDeliveryRes_001(
                                      rc, md_CMN_AC_EVSEStatus_002(
                                      md_CMN_AC_EVSEStatus_001(?,?,?))))))
                                      -> value v_responseMessage {

        tc_V2G_EVCC_Msg_Timer.stop;
        if(v_responseMessage.body.bodyElement.powerDeliveryRes.eVSEStatus.
           aC_EVSEStatus.eVSENotification != none_) {
           logWarn("The field 'eVSENotification' should be set to 'none_'.");
        }
        setverdict(pass, "PowerDeliveryRes message was correct.");
   }
}

altstep a_DC_PowerDeliveryRes_Message(ChargeProgressType c, ResponseCodeType rc)
                                      runs on SECC_Tester {
  var V2G_Message v_responseMessage;
  [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_CMN_PowerDeliveryRes_001(rc,
                                      md_CMN_DC_EVSEStatus_002(
                                      md_CMN_DC_EVSEStatus_001(?,?,*,?))))))
                                      -> value v_responseMessage {

        tc_V2G_EVCC_Msg_Timer.stop;
        if(ispresent(v_responseMessage.body.bodyElement.powerDeliveryRes.
                     eVSEStatus.dC_EVSEStatus.eVSEIsolationStatus)) {
            if(v_responseMessage.body.bodyElement.powerDeliveryRes.eVSEStatus.
               dC_EVSEStatus.eVSEIsolationStatus != valid) {
               logWarn("The field 'eVSEIsolationStatus' should be set to 'valid'.");
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
        }
        if(v_responseMessage.body.bodyElement.powerDeliveryRes.eVSEStatus.
           dC_EVSEStatus.eVSEStatusCode != eVSE_Ready) {
           logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
        }
        if(v_responseMessage.body.bodyElement.powerDeliveryRes.eVSEStatus.
           dC_EVSEStatus.eVSENotification != none_) {
           logWarn("The field 'eVSENotification' should be set to 'none_'.");
        }
        setverdict(pass, "PowerDeliveryRes message was correct.");
    }
}

function f_SECC_AC_TB_VTB_PowerDelivery_001(in ChargeProgressType chargeProgressType,
                                            in HAL_61851_Listener v_HAL_61851_Listener,
                                            in verdicttype v_vct)
                                            runs on SECC_Tester return verdicttype {

    var charstring v_message;
    const integer c_maxProfileEntries := 24;
    const integer c_numberProfilEntries := sizeof(vc_PMaxSchedule.pMaxScheduleEntry_list);
    var integer v_numberProfilEntries;
    if(c_numberProfilEntries > c_maxProfileEntries) {
        v_numberProfilEntries := c_maxProfileEntries;
    } else {v_numberProfilEntries := c_numberProfilEntries; }
    const integer n := v_numberProfilEntries;
    var ProfileEntryType v_profileList[n];

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);

    if(chargeProgressType == start_) {
        for (var integer i:=0; i<n; i:=i + 1) {
            v_profileList[i] :=
            md_CMN_CMN_ProfileEntry_001(vc_PMaxSchedule.pMaxScheduleEntry_list[i].
                                        timeInterval.relativeTimeInterval.start_,
                                        vc_PMaxSchedule.pMaxScheduleEntry_list[i].
                                        pMax, omit);
        }
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_AC_PowerDeliveryReq_003(
                                      chargeProgressType, vc_SAScheduleTupleId,
                                      md_CMN_CMN_ChargingProfile_001(v_profileList)))));
    }
    else {
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_AC_PowerDeliveryReq_001(
                                        chargeProgressType, vc_SAScheduleTupleId))));
    }
    f_SECC_61851_State_CorDorB(chargeProgressType, v_HAL_61851_Listener, false);
    alt {
      [] a_AC_PowerDeliveryRes_Message(chargeProgressType, oK);
      [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                     "terminated by the SUT.");
      [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {powerDeliveryRes := ?});
      [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                   "but cannot be decoded.");
      [] a_SECC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryRes message, " &
                                                    "but with invalid data content.");
      [] a_SECC_Timeout(v_vct);
    }

    return getverdict;
}

function f_SECC_AC_TB_VTB_PowerDelivery_002(in ChargeProgressType chargeProgressType,
                                            in HAL_61851_Listener v_HAL_61851_Listener)
                                            runs on SECC_Tester return verdicttype {

    var charstring v_message;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    f_SECC_61851_State_CorDorB(chargeProgressType, v_HAL_61851_Listener, true);

    alt
    {
     []
     a_Shutdown_Oscillator(pass){
         isShutdownOSC := true;
         if(isClosedTCP == false) {repeat;}
     }
     []    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                          fAILED_SequenceError)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                        "'fAILED_SequenceError' is successful.");

        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));
        isErrorMessage := true;
        repeat;
     }
     [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                   "terminated by the SUT.") {
          isClosedTCP := true;
          tc_CMN_TCP_Connection_Termination_Timer.stop;
          if(isShutdownOSC == false) {
             logWarn("SECC oscillator was not shut down before the " &
                     "TCP connection was terminated.");}
          pt_HAL_61851_Internal_Port.clear;
     }
     [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                    {serviceDiscoveryRes := ?});
     [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                 "but cannot be decoded.");
     [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                                 "but with invalid data content.");
     [] a_SECC_Timeout(fail);
     [] a_SECC_TCP_Connection_Termination_Timeout() {
         v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                         "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
     }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                        "message with responseCode 'fAILED_SequenceError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
function f_SECC_AC_TB_VTB_PowerDelivery_003(in ChargeProgressType chargeProgressType,
                                            in HAL_61851_Listener v_HAL_61851_Listener)
                                            runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(
                                 f_rnd_SessionID(1,429496)),
                                 md_CMN_BodyType_001(
                                 md_CMN_AC_PowerDeliveryReq_001(
                                 chargeProgressType, vc_SAScheduleTupleId))));

    f_SECC_61851_State_CorDorB(chargeProgressType, v_HAL_61851_Listener, true);

    alt{
       []
       a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
       }
       [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(?),
                                            md_CMN_BodyType_001(
                                            md_CMN_CMN_PowerDeliveryRes_001(
                                            fAILED_UnknownSession, ?)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass,"PowerDeliveryRes with responseCode " &
                       "'fAILED_UnknownSession' is successful.");

        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));
        isErrorMessage := true;
        repeat;
     }
     [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                   "terminated by the SUT.") {
          isClosedTCP := true;
          tc_CMN_TCP_Connection_Termination_Timer.stop;
          if(isShutdownOSC == false) {
             logWarn("SECC oscillator was not shut down before the " &
                     "TCP connection was terminated.");}
          pt_HAL_61851_Internal_Port.clear;
     }
     [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                    {powerDeliveryRes := ?});
     [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                 "but cannot be decoded.");
     [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                                 "but with invalid data content.");
     [] a_SECC_Timeout(fail);
     [] a_SECC_TCP_Connection_Termination_Timeout() {
         v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                         "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
     }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a PowerDeliveryRes " &
                       "message with responseCode 'fAILED_UnknownSession' " &
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_AC_TB_VTB_PowerDelivery_004(in ChargeProgressType chargeProgressType,
                                            in HAL_61851_Listener v_HAL_61851_Listener)
                                            runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    const integer v_sizeMax := 24;
    var float v_powerValue := 1000.0;
    var charstring v_powerUnit := "w";
    const integer v_size := sizeof(vc_PMaxSchedule.pMaxScheduleEntry_list);
    var PhysicalValueType pMaxNew;
    var ChargingProfileType v_chargingProfile;

    if(v_size > v_sizeMax) {
        var ProfileEntryType v_profileList1[v_sizeMax];
        for (var integer i:=0; i<v_sizeMax; i:=i + 1) {
            v_profileList1[i] := md_CMN_CMN_ProfileEntry_001(vc_PMaxSchedule.
                                 pMaxScheduleEntry_list[i].timeInterval.relativeTimeInterval.
                                 start_, vc_PMaxSchedule.pMaxScheduleEntry_list[i].pMax, 3);
        }
        pMaxNew := v_profileList1[0].chargingProfileEntryMaxPower;
        pMaxNew := fx_setPhysicalValueType(f_CMN_PhysicalValue_GetValue(
                                       pMaxNew) + v_powerValue, v_powerUnit);
        v_profileList1[0].chargingProfileEntryMaxPower := pMaxNew;
        v_chargingProfile := md_CMN_CMN_ChargingProfile_001(v_profileList1);
    } else {
        var ProfileEntryType v_profileList2[v_size];
        for (var integer i:=0; i<v_size; i:=i + 1) {
            v_profileList2[i] := md_CMN_CMN_ProfileEntry_001(vc_PMaxSchedule.
                                 pMaxScheduleEntry_list[i].timeInterval.relativeTimeInterval.
                                 start_, vc_PMaxSchedule.pMaxScheduleEntry_list[i].pMax, 3);
        }
        pMaxNew := v_profileList2[0].chargingProfileEntryMaxPower;
        pMaxNew := fx_setPhysicalValueType(f_CMN_PhysicalValue_GetValue(
                                       pMaxNew) + v_powerValue, v_powerUnit);
        v_profileList2[0].chargingProfileEntryMaxPower := pMaxNew;
        v_chargingProfile := md_CMN_CMN_ChargingProfile_001(v_profileList2);
    }

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_CMN_AC_PowerDeliveryReq_003(
                                 chargeProgressType, vc_SAScheduleTupleId,
                                 v_chargingProfile))));

    f_SECC_61851_State_CorDorB(chargeProgressType, v_HAL_61851_Listener, true);


    alt
    {
      []
      a_Shutdown_Oscillator(pass){
            isShutdownOSC := true;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_PowerDeliveryRes_001(
                                          fAILED_ChargingProfileInvalid, ?)))) {

         tc_V2G_EVCC_Msg_Timer.stop;
         setverdict(pass,"PowerDeliveryRes with responseCode " &
                            "'fAILED_ChargingProfileInvalid' is successful.");

         pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
         isErrorMessage := true;
         repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                  "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {powerDeliveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                                 "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
      }
      if(not isErrorMessage) {
          v_charstring := v_charstring & "SUT has not sent a PowerDeliveryRes " &
                          "message with responseCode 'fAILED_ChargingProfileInvalid' " &
                          "or other message data content was not conform. ";
      }
      if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
         setverdict(fail,v_charstring);
      }
      return getverdict;
  }

  function f_SECC_AC_TB_VTB_PowerDelivery_005(in ChargeProgressType chargeProgressType,
                                              in HAL_61851_Listener v_HAL_61851_Listener)
                                              runs on SECC_Tester return verdicttype {

      var charstring v_message;
      var boolean isValid;
      const PhysicalValueType power := vc_EVSEMaximumPowerLimit;
      var boolean isShutdownOSC := false;
      var boolean isClosedTCP := false;
      var boolean isErrorMessage := false;
      var charstring v_charstring := "";

      const integer n := 7;
      var urn_iso_15118_2_2013_MsgDataTypes.ProfileEntryType v_profileList[n];
      v_profileList[0] := md_CMN_CMN_ProfileEntry_001(1400, power, 1);
      v_profileList[1] := md_CMN_CMN_ProfileEntry_001(1410, power, 2);
      v_profileList[2] := md_CMN_CMN_ProfileEntry_001(1420, power, 3);
      v_profileList[3] := md_CMN_CMN_ProfileEntry_001(1430, power, 2);
      v_profileList[4] := md_CMN_CMN_ProfileEntry_001(1440, power, 1);
      v_profileList[5] := md_CMN_CMN_ProfileEntry_001(1450, power, 2);
      v_profileList[6] := md_CMN_CMN_ProfileEntry_001(1500, power, 3);

      vc_SAScheduleTupleId := 255;
      isValid := checkInvalidTupleID(vc_SAScheduleTupleId, vc_SASchedule_List);

          f_SECC_setIsConfirmationFlagDC(true);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);

        if(isValid){
            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_AC_PowerDeliveryReq_003(
                                          chargeProgressType, vc_SAScheduleTupleId,
                                          md_CMN_CMN_ChargingProfile_001(v_profileList)))));

            f_SECC_61851_State_CorDorB(chargeProgressType, v_HAL_61851_Listener, true);


            alt
            {
              []
              a_Shutdown_Oscillator(pass){
                    isShutdownOSC := true;
                    if(isClosedTCP == false) {repeat;}
              }
              [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_CMN_PowerDeliveryRes_001(
                                                 fAILED_TariffSelectionInvalid, ?)))) {

               tc_V2G_EVCC_Msg_Timer.stop;
               setverdict(pass,"PowerDeliveryRes with responseCode " &
                                 "'fAILED_TariffSelectionInvalid' is successful.");

               pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
               isErrorMessage := true;
               repeat;
              }
              [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                        "terminated by the SUT.") {
                   isClosedTCP := true;
                   tc_CMN_TCP_Connection_Termination_Timer.stop;
                   if(isShutdownOSC == false) {
                      logWarn("SECC oscillator was not shut down before the " &
                              "TCP connection was terminated.");}
                   pt_HAL_61851_Internal_Port.clear;
              }
              [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {powerDeliveryRes := ?});
              [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                      "but cannot be decoded.");
              [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                                       "but with invalid data content.");
              [] a_SECC_Timeout(fail);
              [] a_SECC_TCP_Connection_Termination_Timeout() {
                  v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                 "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
              }
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a PowerDeliveryRes " &
                            "message with responseCode 'fAILED_TariffSelectionInvalid' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
function f_SECC_AC_TB_VTB_PowerDelivery_006(in ChargeProgressType chargeProgressType,
                                            in HAL_61851_Listener v_HAL_61851_Listener)
                                            runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var IEC_61851_States v_state;

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_AC_PowerDeliveryReq_003(renegotiate,
                                vc_SAScheduleTupleId, omit))));


    alt
    {
     []
     a_Shutdown_Oscillator(pass){
            isShutdownOSC := true;
            if(isClosedTCP == false) {repeat;}
     }
     [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                       md_CMN_MessageHeaderType_001(vc_SessionID),
                                       md_CMN_BodyType_001(
                                       md_CMN_CMN_PowerDeliveryRes_001(fAILED, ?)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass,"PowerDeliveryRes with responseCode 'fAILED' is successful. " &
                        "No PowerDeliveryReq with ChargeProgress " &
                        "set to 'Start' has been received before. ");

        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));
        isErrorMessage := true;
        repeat;
     }
     [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                   "terminated by the SUT.") {
          isClosedTCP := true;
          tc_CMN_TCP_Connection_Termination_Timer.stop;
          if(isShutdownOSC == false) {
             logWarn("SECC oscillator was not shut down before the " &
                     "TCP connection was terminated.");}
          pt_HAL_61851_Internal_Port.clear;
     }
     [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                    {powerDeliveryRes := ?});
     [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                 "but cannot be decoded.");
     [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                                 "but with invalid data content.");
     [] a_SECC_Timeout(fail);
     [] a_SECC_TCP_Connection_Termination_Timeout() {
         v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                         "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
     }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a PowerDeliveryRes " &
                        "message with responseCode 'fAILED' " &
                        "or other message data content was not conform. ";
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

  function f_SECC_DC_TB_VTB_PowerDelivery_001(in ChargeProgressType chargeProgressType,
                                              in HAL_61851_Listener v_HAL_61851_Listener,
                                              in boolean v_chargingComplete,
                                              in boolean v_setState,
                                              in verdicttype v_vct)
                                              runs on SECC_Tester
                                              return verdicttype {

    var charstring v_message;
    var PhysicalValueType v_value;
    const integer c_maxProfileEntries := 24;
    const integer c_numberProfilEntries := sizeof(vc_PMaxSchedule.pMaxScheduleEntry_list);
    var integer v_numberProfilEntries;
    if(c_numberProfilEntries > c_maxProfileEntries) {
        v_numberProfilEntries := c_maxProfileEntries;
    } else {v_numberProfilEntries := c_numberProfilEntries; }
    const integer n := v_numberProfilEntries;
    var ProfileEntryType v_profileList[n];

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);

    if(chargeProgressType == start_) {
        for (var integer i:=0; i<n; i:=i + 1) {
            if(f_CMN_PhysicalValue_GetValue(vc_PMaxSchedule.pMaxScheduleEntry_list[i].pMax)>
               f_CMN_PhysicalValue_GetValue(par_EVMaximumPowerLimit)) {
                v_value := par_EVMaximumPowerLimit;
            } else {
                v_value := vc_PMaxSchedule.pMaxScheduleEntry_list[i].pMax;
            }
            v_profileList[i] :=
            md_CMN_CMN_ProfileEntry_001(vc_PMaxSchedule.pMaxScheduleEntry_list[i].
                                        timeInterval.relativeTimeInterval.start_,
                                        v_value, omit);
        }
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_DC_PowerDeliveryReq_002(chargeProgressType,
                                      vc_SAScheduleTupleId,
                                      md_CMN_CMN_ChargingProfile_001(v_profileList),
                                      md_CMN_DC_EVPowerDeliveryParameter(vc_DC_EVStatus,
                                      omit, v_chargingComplete)))));
    }
    else {

        if(chargeProgressType == stop_ and not PICS_SECC_DC_WeldingDetection){
            vc_DC_EVStatus.eVReady := false;
        }

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_DC_PowerDeliveryReq_002(chargeProgressType,
                                      vc_SAScheduleTupleId, omit,
                                      md_CMN_DC_EVPowerDeliveryParameter(vc_DC_EVStatus,
                                      omit, v_chargingComplete)))));
    }

    alt
    {
      [] a_DC_PowerDeliveryRes_Message(chargeProgressType, oK) {
           if(chargeProgressType != start_ and v_setState) {
            f_SECC_changeValidStateCondition(e_StateB);
             f_SECC_setState(e_StateB, v_HAL_61851_Listener);
            sleep(0.5);
         }
      }
      [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                    "terminated by the SUT.");
    [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                    {powerDeliveryRes := ?});
    [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                          "but cannot be decoded.");
    [] a_SECC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryRes message, " &
                                                "but with invalid data content.");
    [] a_SECC_Timeout(v_vct);
  }
  return getverdict;
}

function f_SECC_DC_TB_VTB_PowerDelivery_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

          f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt
    {
       []
       a_Shutdown_Oscillator(pass){
            isShutdownOSC := true;
            if(isClosedTCP == false) {repeat;}
       }
       [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                          fAILED_SequenceError)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                        "'fAILED_SequenceError' is successful.");

        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                     e_getPortStatus));
        isErrorMessage := true;
        repeat;
      }
      [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                    "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
           if(isShutdownOSC == false) {
              logWarn("SECC oscillator was not shut down before the " &
                      "TCP connection was terminated.");}
           pt_HAL_61851_Internal_Port.clear;
      }
      [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                     {serviceDiscoveryRes := ?});
      [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                            "but cannot be decoded.");
      [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                                 "but with invalid data content.");
      [] a_SECC_Timeout(fail);
      [] a_SECC_TCP_Connection_Termination_Timeout() {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                        "message with responseCode 'fAILED_SequenceError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_PowerDelivery_003(in ChargeProgressType chargeProgressType)
                                            runs on SECC_Tester return verdicttype {

    var charstring v_message;
    const PhysicalValueType power := vc_EVSEMaximumPowerLimit;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

          f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(chargeProgressType == stop_ and not PICS_SECC_DC_WeldingDetection){
        vc_DC_EVStatus.eVReady := false;
    }

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(
                                  f_rnd_SessionID(1,429496)),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_PowerDeliveryReq_002(
                                  chargeProgressType, vc_SAScheduleTupleId, omit,
                                  md_CMN_DC_EVPowerDeliveryParameter(
                                  vc_DC_EVStatus, omit, false)))));

    alt
    {
       []
       a_Shutdown_Oscillator(pass){
            isShutdownOSC := true;
            if(isClosedTCP == false) {repeat;}
       }
       [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(?),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_PowerDeliveryRes_001(
                                        fAILED_UnknownSession, ?)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass,"PowerDeliveryRes with responseCode " &
                        "'fAILED_UnknownSession' is successful.");

        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));
        isErrorMessage := true;
        repeat;
      }
      [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                              "terminated by the SUT.") {
          isClosedTCP := true;
          tc_CMN_TCP_Connection_Termination_Timer.stop;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
          if(isShutdownOSC == false) {
             logWarn("SECC oscillator was not shut down before the " &
                     "TCP connection was terminated.");}
          pt_HAL_61851_Internal_Port.clear;
      }
      [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                       {powerDeliveryRes := ?});
      [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                            "but cannot be decoded.");
      [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                              "but with invalid data content.");
      [] a_SECC_Timeout(fail);
      [] a_SECC_TCP_Connection_Termination_Timeout() {
          v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                          "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
      }
     }
     if(not isErrorMessage) {
         v_charstring := v_charstring & "SUT has not sent a PowerDeliveryRes " &
                         "message with responseCode 'fAILED_UnknownSession' " &
                         "or other message data content was not conform. ";
     }
     if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
        setverdict(fail,v_charstring);
     }
     return getverdict;
}

function f_SECC_DC_TB_VTB_PowerDelivery_004(in ChargeProgressType chargeProgressType)
                                            runs on SECC_Tester return verdicttype {

     var charstring v_message;
     var boolean isShutdownOSC := false;
     var boolean isClosedTCP := false;
     var boolean isErrorMessage := false;
     var charstring v_charstring := "";
     const integer v_sizeMax := 24;
     var float v_powerValue := 1000.0;
     var charstring v_powerUnit := "w";
     const integer v_size := sizeof(vc_PMaxSchedule.pMaxScheduleEntry_list);
     var PhysicalValueType pMaxNew;
     var ChargingProfileType v_chargingProfile;

     if(v_size > v_sizeMax) {
         var ProfileEntryType v_profileList1[v_sizeMax];
         for (var integer i:=0; i<v_sizeMax; i:=i + 1) {
             v_profileList1[i] := md_CMN_CMN_ProfileEntry_001(vc_PMaxSchedule.
                                  pMaxScheduleEntry_list[i].timeInterval.relativeTimeInterval.
                                  start_, vc_PMaxSchedule.pMaxScheduleEntry_list[i].pMax, omit);
         }
         pMaxNew := v_profileList1[0].chargingProfileEntryMaxPower;
         pMaxNew := fx_setPhysicalValueType(f_CMN_PhysicalValue_GetValue(
                                            pMaxNew) + v_powerValue, v_powerUnit);
         v_profileList1[0].chargingProfileEntryMaxPower := pMaxNew;
         v_chargingProfile := md_CMN_CMN_ChargingProfile_001(v_profileList1);
     } else {
         var ProfileEntryType v_profileList2[v_size];
         for (var integer i:=0; i<v_size; i:=i + 1) {
             v_profileList2[i] := md_CMN_CMN_ProfileEntry_001(vc_PMaxSchedule.
                                  pMaxScheduleEntry_list[i].timeInterval.relativeTimeInterval.
                                  start_, vc_PMaxSchedule.pMaxScheduleEntry_list[i].pMax, omit);
         }
         pMaxNew := v_profileList2[0].chargingProfileEntryMaxPower;
         pMaxNew := fx_setPhysicalValueType(f_CMN_PhysicalValue_GetValue(
                                            pMaxNew) + v_powerValue, v_powerUnit);
         v_profileList2[0].chargingProfileEntryMaxPower := pMaxNew;
         v_chargingProfile := md_CMN_CMN_ChargingProfile_001(v_profileList2);
     }

           f_SECC_setIsConfirmationFlagDC(true);
         f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                          vc_FrequencyOff);
         f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                          vc_DutyCycleOscOff);
         deactivate(vc_Default_IEC_61851_ListenerBehavior);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    if(chargeProgressType == stop_ and not PICS_SECC_DC_WeldingDetection){
        vc_DC_EVStatus.eVReady := false;
    }

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_PowerDeliveryReq_002(
                                chargeProgressType, vc_SAScheduleTupleId,
                                v_chargingProfile,
                                md_CMN_DC_EVPowerDeliveryParameter(
                                vc_DC_EVStatus, omit, false)))));

    alt
    {
      []
      a_Shutdown_Oscillator(pass){
            isShutdownOSC := true;
            if(isClosedTCP == false) {repeat;}
      }
      [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_PowerDeliveryRes_001(
                                        fAILED_ChargingProfileInvalid, ?)))) {

       tc_V2G_EVCC_Msg_Timer.stop;
       setverdict(pass,"PowerDeliveryRes with responseCode " &
                        "'fAILED_ChargingProfileInvalid' is successful.");

       pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));
       isErrorMessage := true;
       repeat;
      }
      [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                  "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
           if(isShutdownOSC == false) {
              logWarn("SECC oscillator was not shut down before the " &
                      "TCP connection was terminated.");}
           pt_HAL_61851_Internal_Port.clear;
      }
      [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                    {powerDeliveryRes := ?});
      [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                 "but cannot be decoded.");
      [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                                "but with invalid data content.");
      [] a_SECC_Timeout(fail);
      [] a_SECC_TCP_Connection_Termination_Timeout() {
          v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                          "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
      }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a PowerDeliveryRes " &
                        "message with responseCode 'fAILED_ChargingProfileInvalid' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_PowerDelivery_005(in ChargeProgressType chargeProgressType)
                                            runs on SECC_Tester return verdicttype {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var charstring v_message;
var boolean isValid;
const PhysicalValueType power := vc_EVSEMaximumPowerLimit;
var boolean isShutdownOSC := false;
var boolean isClosedTCP := false;
var boolean isErrorMessage := false;
var charstring v_charstring := "";

const integer n := 7;
var urn_iso_15118_2_2013_MsgDataTypes.ProfileEntryType v_profileList[n];
v_profileList[0] := md_CMN_CMN_ProfileEntry_001(1400, power, 1);
v_profileList[1] := md_CMN_CMN_ProfileEntry_001(1410, power, 2);
v_profileList[2] := md_CMN_CMN_ProfileEntry_001(1420, power, 3);
v_profileList[3] := md_CMN_CMN_ProfileEntry_001(1430, power, 2);
v_profileList[4] := md_CMN_CMN_ProfileEntry_001(1440, power, 1);
v_profileList[5] := md_CMN_CMN_ProfileEntry_001(1450, power, 2);
v_profileList[6] := md_CMN_CMN_ProfileEntry_001(1500, power, 3);

vc_SAScheduleTupleId := 255;
isValid := checkInvalidTupleID(vc_SAScheduleTupleId, vc_SASchedule_List);

      f_SECC_setIsConfirmationFlagDC(true);
    f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                     vc_FrequencyOff);
    f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                     vc_DutyCycleOscOff);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

if(chargeProgressType == stop_ and not PICS_SECC_DC_WeldingDetection){
    vc_DC_EVStatus.eVReady := false;
}

if(isValid){
    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_PowerDeliveryReq_002(
                                chargeProgressType, vc_SAScheduleTupleId,
                                md_CMN_CMN_ChargingProfile_001(v_profileList),
                                md_CMN_DC_EVPowerDeliveryParameter(
                                vc_DC_EVStatus, omit, false)))));

    alt
    {
      []
      a_Shutdown_Oscillator(pass){
            isShutdownOSC := true;
            if(isClosedTCP == false) {repeat;}
      }
      [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_PowerDeliveryRes_001(
                                        fAILED_TariffSelectionInvalid, ?)))) {

       tc_V2G_EVCC_Msg_Timer.stop;
       setverdict(pass,"PowerDeliveryRes with responseCode " &
                        "'fAILED_TariffSelectionInvalid' is successful.");

       pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));
       isErrorMessage := true;
       repeat;
      }
      [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                    "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
           if(isShutdownOSC == false) {
              logWarn("SECC oscillator was not shut down before the " &
                      "TCP connection was terminated.");}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {powerDeliveryRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                  "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                                 "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a PowerDeliveryRes " &
                        "message with responseCode 'fAILED_TariffSelectionInvalid' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_PowerDelivery_006() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

          f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_PowerDeliveryReq_002(
                                renegotiate,vc_SAScheduleTupleId, omit,
                                md_CMN_DC_EVPowerDeliveryParameter(
                                vc_DC_EVStatus, omit, false)))));

    alt {
      []
      a_Shutdown_Oscillator(pass){
            isShutdownOSC := true;
            if(isClosedTCP == false) {repeat;}
      }
      [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_PowerDeliveryRes_001(fAILED, ?)))) {

        tc_V2G_EVCC_Msg_Timer.stop;
        setverdict(pass,"PowerDeliveryRes with responseCode 'fAILED' is successful. " &
                        "No PowerDeliveryReq with ChargeProgress " &
                        "set to 'Start' has been received before. ");

        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                      e_getPortStatus));
        isErrorMessage := true;
        repeat;
      }
      [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                    "terminated by the SUT.") {
          isClosedTCP := true;
          tc_CMN_TCP_Connection_Termination_Timer.stop;
          if(isShutdownOSC == false) {
             logWarn("SECC oscillator was not shut down before the " &
                     "TCP connection was terminated.");}
          pt_HAL_61851_Internal_Port.clear;
      }
      [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                      {powerDeliveryRes := ?});
      [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                            "but cannot be decoded.");
      [] a_SECC_Unexpected_Message_Content(fail, "Received PowerDeliveryRes message, " &
                                                  "but with invalid data content.");
      [] a_SECC_Timeout(fail);
      [] a_SECC_TCP_Connection_Termination_Timeout() {
          v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                          "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
      }
     }
     if(not isErrorMessage) {
         v_charstring := v_charstring & "SUT has not sent a PowerDeliveryRes " &
                         "message with responseCode 'fAILED' " &
                         "or other message data content was not conform. ";
     }
     if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
        setverdict(fail,v_charstring);
     }
     return getverdict;
}

function f_SECC_CMN_TB_VTB_PowerDelivery_001() runs on SECC_Tester
                                                return verdicttype {

  timer t_wait;
  timer t_sequenceTimeout;
  var boolean isShutdownOSC := false;
  var boolean isClosedTCP := false;
  var float v_timeShift := 1.0;

  // Wait until sequence timeout in SUT should expire
  vc_processingTime := tc_V2G_EVCC_PreCharge_Timer.read;
  t_wait.start(par_V2G_EVCC_PreCharge_Timeout - vc_processingTime - v_timeShift);
  t_sequenceTimeout.start(par_V2G_SECC_Sequence_Timeout);
  pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

  alt {
      [] t_wait.timeout;
      [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                    "terminated by the SUT " &
                                                    "before timeout has expired.");
      [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
          setverdict(fail, "No V2G message was expected.");
      }
  }

  if(getverdict == pass) {

          f_SECC_setIsConfirmationFlagDC(true);
          f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                           vc_FrequencyOff);
          f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                           vc_DutyCycleOscOff);
          deactivate(vc_Default_IEC_61851_ListenerBehavior);

      sleep(v_timeShift - par_CMN_Transmission_Delay);

      vc_processingTime := t_sequenceTimeout.read;
      tc_CMN_TCP_Connection_Termination_Timer.start(
      par_CMN_TCP_Connection_Termination_Timeout +
      (par_V2G_SECC_Sequence_Timeout
      - vc_processingTime - par_CMN_Transmission_Delay));

      alt {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
  }

  function checkInvalidTupleID(SAIDType v_id, SAScheduleListType
                               vc_SAScheduleTuple_List)
                               return boolean {

    for (var integer i:=0; i<sizeof(vc_SAScheduleTuple_List); i:=i + 1) {
        if(v_id == vc_SAScheduleTuple_List.sAScheduleTuple_list[i].sAScheduleTupleID) {
            setverdict(inconc,"Selected tupleID is not invalid.");
            return false;
        }
    }
    return true;
  }
}
```

I.2.2.9 SECC functions for CertificateUpdate

```
module TestBehavior_SECC_CertificateUpdate {

    import from Timer_15118_2 all;
    import from Pics_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_SECC_CertificateUpdate all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_SECC_ServiceDiscovery all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Services_HAL_61851_1 all;
    import from TestBehavior_SECC_CertificateInstallation all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Services_Security all;
    import from http_www_w3_org_2000_09_xmldsig all;
    import from Templates_CMN_Security all;
    import from LibFunctions_15118_2 all;
    import from urn_iso_15118_2_2013_X509Def all;
    import from DataStructure_PICS_15118_2 all;
    import from Timer_15118 all;

    function f_SECC_CMN_TB_VTB_CertificateUpdate_001(in verdicttype v_vct)
                                                     runs on SECC_Tester
                                                     return verdicttype {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var V2G_Message v_responseMessage;
var charstring v_message;
var boolean isSignatureOk := true;
var boolean isCertChainOk := false;

if(f_checkCertificateService(vc_serviceList)) {

    var X509CertificateType v_leafCert :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_contract_leaf_expires);
    var X509CertificateType v_subCert2 :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub2_valid);
    var X509CertificateType v_subCert1 :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub1_valid);

    if(getverdict == pass) {
        var charstring v_EMAID := fx_getEMAID(v_leafCert);
        vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
        vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.
                               tBSCertificate.serial_Number));

        var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
                                            md_SECC_CMN_CertificateUpdateReq_001(
                                            "cuReq",
                                            md_CertificateChainType_002(
                                            v_leafCert, v_subCert2, v_subCert1),
                                            v_EMAID, vc_x509IssuerName,
                                            vc_x509SerialNumber));

        var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                   cc_crt_contract_leaf_expires);

        var Signature aSignature := md_CMN_Signature_001(
                                 md_CMN_SignedInfoType_001("cuReq",
                                 f_generateDigestFromBodyType(
                                 myCertificateUpdateBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_003(
                                   vc_SessionID, aSignature),
                                   myCertificateUpdateBodyType));

       alt  {
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_003(vc_SessionID,
                                        md_CMN_Signature_002(
                                        mw_CMN_SignedInfoType_003)),
                                        md_CMN_BodyType_001(
                                        mdw_SECC_CMN_CertificateUpdateRes_001(
                                        m_CMN_sAProvisioningCertificateChain_001(),
                                        m_CMN_contractSignatureCertChain_001(),
                                        v_EMAID)))) -> value v_responseMessage {

         tc_V2G_EVCC_Msg_Timer.stop;

         vc_sAProvisioningCertChain := v_responseMessage.body.bodyElement.
                                            certificateUpdateRes.
                                            sAProvisioningCertificateChain;
         vc_contractSignatureCertChain := v_responseMessage.body.bodyElement.
                                        certificateUpdateRes.
                                        contractSignatureCertChain;
         vc_contractSignatureEncryptedPrivateKey := v_responseMessage.body.
                                                    bodyElement.
                                                    certificateUpdateRes.
                                                    contractSignatureEncryptedPrivateKey;
         vc_dHpublickey := v_responseMessage.body.bodyElement.
                           certificateUpdateRes.dHpublickey;

         vc_eMAID := v_responseMessage.body.bodyElement.
                     certificateUpdateRes.eMAID;

         if(getverdict == pass) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var Signature authSignature := v_responseMessage.header.signature_;

var BodyType certificateUpdateResBodyType := v_responseMessage.
                                             body;

f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                            cc_algorithm_method_2, cc_algorithm_method_3,
                            v_vct);

if(getverdict == pass) {

        isSignatureOk := f_verifySignature(
                         certificateUpdateResBodyType,
                         authSignature,
                         vc_sAProvisioningCertChain.
                         x509Certificate, v_vct);

    if(isSignatureOk) {

        isCertChainOk := fx_verifyCertChain(vc_sAProvisioningCertChain,
                                            vc_v2gRootCert, "CPS");
     } else {
        isCertChainOk := true;
     }

     if(not isCertChainOk) {
            setverdict(v_vct, "CertChain of the oEMProvisioningCert " &
                              "is not valid.");
     }
    if(isCertChainOk) {
        var boolean result := fx_checkCertificateValidity(
                              vc_contractSignatureCertChain.
                              x509Certificate.tBSCertificate.
                              validity.not_Before,
                              vc_contractSignatureCertChain.
                              x509Certificate.tBSCertificate.
                              validity.not_After);

        if(not result) {
                setverdict(v_vct,"Invalid Certificate validity " &
                                 "was detected.");
        }
        if(getverdict == pass) {

         log("EMAID: " & toString<charstring>(vc_eMAID.base));
         result := fx_validateEMAID(vc_eMAID.base);
         if(not result) {
                setverdict(v_vct,"Invalid EMAID syntax was detected.");
         }
         else {
             setverdict(pass, "CertificateUpdateRes message was correct.");
         }

         vc_sharedSecret := fx_calculateSharedSecret(cc_crt_contract_leaf_expires,
                                                     vc_dHpublickey);

        vc_privateKey := fx_decryptContractPrivateKey(vc_sharedSecret,
                         vc_contractSignatureEncryptedPrivateKey);

        //check contract cert private key
        // with given NEW contract cert (public key)
        var boolean isValidContractPrivateKey :=
                                        fx_isPrivateKeyValidForCertificate(
                                        vc_privateKey,
                                        vc_contractSignatureCertChain.
                                        x509Certificate);

        if (not isValidContractPrivateKey) {
                setverdict(v_vct, "Contract private key does not match the " &
                                  "contract certificate.");
        } else {
            fx_storeContractCertificateChain(
                                  vc_contractSignatureCertChain,
                                  vc_privateKey);
            log("VALID contract private key to contract certificate: ",
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                isValidContractPrivateKey);
                            }
                          }
                        }
                      }
                    }
                  }
                  [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                                 "was misleadingly " &
                                                                 "terminated by the SUT.");
                  [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                                   {certificateUpdateRes := ?});
                  [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                               "but cannot be decoded.");
                  [] a_SECC_Unexpected_Message_Content(v_vct, "Received CertificateUpdateRes" &
                                                              " message, but with invalid " &
                                                              "data content.");
                  [] a_SECC_Timeout(v_vct);
                 }
             }
        }
        else {setverdict(inconc,"Wrong PIXIT selection. No certificate service has been offered.");}
        return getverdict;
    }

function f_SECC_CMN_TB_VTB_CertificateUpdate_002() runs on SECC_Tester
                                                   return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        []    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                          fAILED_SequenceError)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                            "'fAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
            pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {serviceDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                              "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes message, " &
                                             "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                        "message with responseCode 'fAILED_SequenceError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
  }

function f_SECC_CMN_TB_VTB_CertificateUpdate_003() runs on SECC_Tester
                                                   return verdicttype {

     var charstring v_message;
     var boolean isShutdownOSC := false;
     var boolean isClosedTCP := false;
     var boolean isErrorMessage := false;
     var charstring v_charstring := "";

         f_SECC_setIsConfirmationFlagDC(true);
         f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                          vc_FrequencyOff);
         f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                          vc_DutyCycleOscOff);
         deactivate(vc_Default_IEC_61851_ListenerBehavior);

     var X509CertificateType v_leafCert :=
     f_getCertificate(vc_IdCertificateMap, cc_crt_contract_leaf_expires);
     var X509CertificateType v_subCert2 :=
     f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub2_valid);
     var X509CertificateType v_subCert1 :=
     f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub1_valid);

     if(getverdict == pass) {
         var charstring v_EMAID := fx_getEMAID(v_leafCert);
         vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
         vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

         var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
                                                 md_SECC_CMN_CertificateUpdateReq_001(
                                                 "cuReq",
                                                 md_CertificateChainType_002(v_leafCert,
                                                 v_subCert2, v_subCert1),
                                                 v_EMAID, vc_x509IssuerName,
                                                 vc_x509SerialNumber));

         var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                        cc_crt_contract_leaf_expires);

         var Signature aSignature := md_CMN_Signature_001(
                                     md_CMN_SignedInfoType_001("cuReq",
                                     f_generateDigestFromBodyType(
                                     myCertificateUpdateBodyType)), v_privateKey);

         tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);
         tc_CMN_TCP_Connection_Termination_Timer.start(
         par_CMN_TCP_Connection_Termination_Timeout);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_003(
                                    f_rnd_SessionID(1,429496), aSignature),
                                    myCertificateUpdateBodyType));

       alt  {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_002(?),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_CertificateUpdateRes_002(
                                          fAILED_UnknownSession)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"CertificateUpdateRes with responseCode " &
                               "'fAILED_UnknownSession' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                         e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {certificateUpdateRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
                                                       "message, but with invalid " &
                                                       "data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
       }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a CertificateUpdateRes " &
                        "message with responseCode 'fAILED_UnknownSession' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
  }

 function f_SECC_CMN_TB_VTB_CertificateUpdate_004() runs on SECC_Tester
                                                    return verdicttype {

     timer t_wait;
     var boolean isShutdownOSC := false;
     var boolean isClosedTCP := false;
     var float v_timeShift := 1.0;

     // Wait until sequence timeout in SUT should expire
     t_wait.start(par_V2G_SECC_Sequence_Timeout -
                  v_timeShift);
     pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

     alt {
         [] t_wait.timeout;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                        "terminated by the SUT " &
                                        "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                        vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                        vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                    "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                 "but TCP connection was " &
                                                 ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_CertificateUpdate_005() runs on SECC_Tester
                                                   return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                        vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                        vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    var X509CertificateType v_leafCert :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_contract_leaf_expires);
    var X509CertificateType v_subCert2 :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub2_valid);
    var X509CertificateType v_subCert1 :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub1_valid);

    if(getverdict == pass) {
        var charstring v_EMAID := fx_getEMAID(v_leafCert);
        vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

        var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
                                                md_SECC_CMN_CertificateUpdateReq_001(
                                                "cuReq",
                                                md_CertificateChainType_002(v_leafCert,
                                                v_subCert2, v_subCert1),
                                                v_EMAID, vc_x509IssuerName,
                                                vc_x509SerialNumber));

        // load wrong private key
        var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                       cc_crt_oem_leaf_valid);

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("cuReq",
                                    f_generateDigestFromBodyType(
                                    myCertificateUpdateBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_003(
                                      vc_SessionID, aSignature),
                                      myCertificateUpdateBodyType));

       alt  {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_002(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 mdw_SECC_CMN_CertificateUpdateRes_002(
                                                 fAILED_SignatureError)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"CertificateUpdateRes with responseCode " &
                                "'fAILED_SignatureError' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {certificateUpdateRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
                                                       "message, but with invalid " &
                                                       "data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
       }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a CertificateUpdateRes " &
                        "message with responseCode 'fAILED_SignatureError' " &
                        "or other message data content was not conform. ";
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      }
      if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
         setverdict(fail,v_charstring);
      }
      return getverdict;
   }

function f_SECC_CMN_TB_VTB_CertificateUpdate_006() runs on SECC_Tester
                                                   return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    var X509CertificateType v_leafCert :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_contract_leaf_expires);
    var X509CertificateType v_subCert2 :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub2_valid);
    var X509CertificateType v_subCert1 :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub1_valid);

    if(getverdict == pass) {
        var charstring v_EMAID := fx_getEMAID(v_leafCert);
        vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
        vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

        var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
                                                     md_SECC_CMN_CertificateUpdateReq_001(
                                                     "cuReq",
                                                     md_CertificateChainType_002(v_leafCert,
                                                     v_subCert2, v_subCert1),
                                                     v_EMAID, vc_x509IssuerName,
                                                     vc_x509SerialNumber));

        var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                       cc_crt_contract_leaf_expires);

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("cuReq",
                                    f_generateDigestFromBodyType(
                                    myCertificateUpdateBodyType)), v_privateKey);

        // change id after signature generation
        myCertificateUpdateBodyType.bodyElement.
        certificateUpdateReq.id := "contentChanged";

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_003(
                                      vc_SessionID, aSignature),
                                      myCertificateUpdateBodyType));

       alt  {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_002(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            mdw_SECC_CMN_CertificateUpdateRes_002(
                                            fAILED_SignatureError)))) {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"CertificateUpdateRes with responseCode " &
                                "'fAILED_SignatureError' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                              e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {certificateUpdateRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
                                                       "message, but with invalid " &
                                                       "data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a CertificateUpdateRes " &
                        "message with responseCode 'fAILED_SignatureError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
 }

function f_SECC_CMN_TB_VTB_CertificateUpdate_007(charstring v_config) runs on SECC_Tester
                                                                    return verdicttype {

    var charstring v_message;
    var PkiType v_wrongPki := PKI_2;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var X509CertificateType v_leafCert;
    var X509CertificateType v_subCert1;
    var X509CertificateType v_subCert2;
    var charstring v_config1 := "WrongSubCertOrder";
    var charstring v_config2 := "WrongPKILeafCert";
    var charstring v_config3 := "WrongPKISub2Cert";
    var charstring v_config4 := "WrongPKISub1Cert";
    var charstring v_config5 := "WrongPKICertChain";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    if(v_config == v_config1) {
        v_leafCert := f_getCertificate(vc_IdCertificateMap, cc_crt_contract_leaf_expires);
        v_subCert2 := f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub1_valid);
        v_subCert1 := f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub2_valid);
    } else if(v_config == v_config2) {
        v_leafCert := fx_loadCertificate(v_wrongPki, cc_crt_contract_leaf_expires);
        v_subCert2 := f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub2_valid);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        v_subCert1 := f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub1_valid);
    } else if(v_config == v_config3) {
        v_leafCert := f_getCertificate(vc_IdCertificateMap, cc_crt_contract_leaf_expires);
        v_subCert2 := fx_loadCertificate(v_wrongPki, cc_crt_mo_sub2_valid);
        v_subCert1 := f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub1_valid);
    } else if(v_config == v_config4) {
        v_leafCert := f_getCertificate(vc_IdCertificateMap, cc_crt_contract_leaf_expires);
        v_subCert2 := f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub2_valid);
        v_subCert1 := fx_loadCertificate(v_wrongPki, cc_crt_mo_sub1_valid);
    } else if(v_config == v_config5) {
        v_leafCert := fx_loadCertificate(v_wrongPki, cc_crt_contract_leaf_expires);
        v_subCert2 := fx_loadCertificate(v_wrongPki, cc_crt_mo_sub2_valid);
        v_subCert1 := fx_loadCertificate(v_wrongPki, cc_crt_mo_sub1_valid);
    } else {setverdict(inconc, "Unknown configuration was detected.");}

    if(getverdict == pass) {
        var charstring v_EMAID := fx_getEMAID(v_leafCert);
        vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
        vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

        var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
                                                     md_SECC_CMN_CertificateUpdateReq_001(
                                                     "cuReq",
                                                     md_CertificateChainType_002(v_leafCert,
                                                     v_subCert2, v_subCert1),
                                                     v_EMAID, vc_x509IssuerName,
                                                     vc_x509SerialNumber));

        var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                       cc_crt_contract_leaf_expires);

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("cuReq",
                                    f_generateDigestFromBodyType(
                                    myCertificateUpdateBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_003(
                                      vc_SessionID, aSignature),
                                      myCertificateUpdateBodyType));

       alt  {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_002(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            mdw_SECC_CMN_CertificateUpdateRes_002(
                                            fAILED_CertChainError)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"CertificateUpdateRes with responseCode " &
                                "'fAILED_CertChainError' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                {certificateUpdateRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
                                                       "message, but with invalid " &
                                                       "data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a CertificateUpdateRes " &
                        "message with responseCode 'fAILED_CertChainError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
  }

function f_SECC_CMN_TB_VTB_CertificateUpdate_008(charstring v_EMAID_Unknown)
                                                 runs on SECC_Tester
                                                 return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var ResponseCodeType v_responseCodeReceived;
    var V2G_Message v_responseMessage;

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    var X509CertificateType v_leafCert :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_contract_leaf_expires);
    var X509CertificateType v_subCert2 :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub2_valid);
    var X509CertificateType v_subCert1 :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub1_valid);
    var charstring v_EMAID := fx_getEMAID(v_leafCert);

    if(v_EMAID != v_EMAID_Unknown) {

        v_EMAID := v_EMAID_Unknown;
        vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
        vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

        var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
                                                 md_SECC_CMN_CertificateUpdateReq_001(
                                                 "cuReq",
                                                 md_CertificateChainType_002(v_leafCert,
                                                 v_subCert2, v_subCert1),
                                                 v_EMAID, vc_x509IssuerName,
                                                 vc_x509SerialNumber));

        var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                       cc_crt_contract_leaf_expires);

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("cuReq",
                                    f_generateDigestFromBodyType(
                                    myCertificateUpdateBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_003(
                                    vc_SessionID, aSignature),
                                    myCertificateUpdateBodyType));

       alt  {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_002(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_CertificateUpdateRes_002(
                                          m_CMN_failedResponseCodeList_002()))))
                                          -> value v_responseMessage {
                tc_V2G_EVCC_Msg_Timer.stop;
                v_responseCodeReceived := v_responseMessage.body.bodyElement.
                                       certificateUpdateRes.responseCode;
                setverdict(pass,"CertificateUpdateRes with responseCode '" &
                               toString<ResponseCodeType>(v_responseCodeReceived) &
                                "' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                         e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                         {certificateUpdateRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
                                                       "message, but with invalid " &
                                                       "data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
       }
       if(not isErrorMessage) {
           v_charstring := v_charstring & "SUT has not sent a CertificateUpdateRes " &
                           "message with corresponding responseCode " &
                           "or other message data content was not conform. ";
       }
       if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
          setverdict(fail,v_charstring);
       }
   } else {
           setverdict(inconc, "Unknown EMAID was valid for the Contract Leaf Certificate.");
   }
   return getverdict;
 }

function f_SECC_CMN_TB_VTB_CertificateUpdate_009(charstring v_certName)
                                                 runs on SECC_Tester
                                                 return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var charstring v_charstring := "";
var ResponseCodeType v_responseCodeReceived;
var V2G_Message v_responseMessage;

    f_SECC_setIsConfirmationFlagDC(true);
    f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                     vc_FrequencyOff);
    f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                     vc_DutyCycleOscOff);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);
var X509CertificateType v_leafCertUnused :=
f_getCertificate(vc_IdCertificateMap, cc_crt_contract_leaf_expires);
var charstring v_EMAID := fx_getEMAID(v_leafCertUnused);
var X509CertificateType v_leafCert :=
f_getCertificate(vc_IdCertificateMap, v_certName);
var X509CertificateType v_subCert2 :=
f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub2_valid);
var X509CertificateType v_subCert1 :=
f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub1_valid);
var charstring v_EMAID_Unknown := fx_getEMAID(v_leafCert);

if(v_EMAID != v_EMAID_Unknown) {

    vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
    vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

    var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
                                          md_SECC_CMN_CertificateUpdateReq_001(
                                          "cuReq",
                                          md_CertificateChainType_002(v_leafCert,
                                          v_subCert2, v_subCert1),
                                          v_EMAID, vc_x509IssuerName,
                                          vc_x509SerialNumber));

    var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                   v_certName);

    var Signature aSignature := md_CMN_Signature_001(
                                md_CMN_SignedInfoType_001("cuReq",
                                f_generateDigestFromBodyType(
                                myCertificateUpdateBodyType)), v_privateKey);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_003(
                                  vc_SessionID, aSignature),
                                  myCertificateUpdateBodyType));

   alt  {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_002(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        mdw_SECC_CMN_CertificateUpdateRes_002(
                                        m_CMN_failedResponseCodeList_002()))))
                                        -> value v_responseMessage {
            tc_V2G_EVCC_Msg_Timer.stop;
            v_responseCodeReceived := v_responseMessage.body.bodyElement.
                                      certificateUpdateRes.responseCode;
            setverdict(pass,"CertificateUpdateRes with responseCode '" &
                            toString<ResponseCodeType>(v_responseCodeReceived) &
                             "' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                             {certificateUpdateRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
                                                       "message, but with invalid " &
                                                       "data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
             v_charstring := v_charstring & "SUT has not sent a CertificateUpdateRes " &
                             "message with corresponding responseCode " &
                             "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    } else {
         setverdict(inconc, "Unknown EMAID was valid for the Contract Leaf Certificate.");
    }
    return getverdict;
  }

function f_SECC_CMN_TB_VTB_CertificateUpdate_010(in charstring v_certName,
                                                 in template ResponseCodeType
                                                 v_responseCode)
                                                 runs on SECC_Tester
                                                 return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var ResponseCodeType v_responseCodeReceived;
    var V2G_Message v_responseMessage;

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    var X509CertificateType v_leafCert :=
    f_getCertificate(vc_IdCertificateMap, v_certName);
    var X509CertificateType v_subCert2 :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub2_valid);
    var X509CertificateType v_subCert1 :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_mo_sub1_valid);

    if(getverdict == pass) {
        var charstring v_EMAID := fx_getEMAID(v_leafCert);
        vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
        vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.tBSCertificate.serial_Number));

        var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
                                                 md_SECC_CMN_CertificateUpdateReq_001(
                                                 "cuReq",
                                                 md_CertificateChainType_002(v_leafCert,
                                                 v_subCert2, v_subCert1),
                                                 v_EMAID, vc_x509IssuerName,
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                vc_x509SerialNumber));

        var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                       v_certName);

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("cuReq",
                                    f_generateDigestFromBodyType(
                                    myCertificateUpdateBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_003(
                                      vc_SessionID, aSignature),
                                      myCertificateUpdateBodyType));

       alt  {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_002(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            mdw_SECC_CMN_CertificateUpdateRes_002(
                                            v_responseCode))))
                                            -> value v_responseMessage {
                tc_V2G_EVCC_Msg_Timer.stop;
                v_responseCodeReceived := v_responseMessage.body.bodyElement.
                                          certificateUpdateRes.responseCode;
                setverdict(pass,"CertificateUpdateRes with responseCode '" &
                                toString<ResponseCodeType>(v_responseCodeReceived) &
                                 "' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                             {certificateUpdateRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateUpdateRes " &
                                                       "message, but with invalid " &
                                                       "data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
       }
    }
    if(not isErrorMessage) {
         v_charstring := v_charstring & "SUT has not sent a CertificateUpdateRes " &
                         "message with corresponding responseCode " &
                         "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      }
}
```

I.2.2.10 SECC functions for CertificateInstallation

```
module TestBehavior_SECC_CertificateInstallation {

    import from Timer_15118_2 all;
    import from Pics_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_SECC_CertificateInstallation all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_ServiceDiscovery all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Services_HAL_61851_1 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Services_Security all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from http_www_w3_org_2000_09_xmldsig all;
    import from Templates_CMN_Security all;
    import from LibFunctions_15118_2 all;
    import from urn_iso_15118_2_2013_X509Def all;
    import from Timer_15118 all;

    function f_SECC_CMN_TB_VTB_CertificateInstallation_001(in verdicttype v_vct)
                                                           runs on SECC_Tester
                                                           return verdicttype {

        var V2G_Message v_responseMessage;
        var charstring v_message;
        var boolean isSignatureOk := true;
        var boolean isCertChainOk := false;

        if(f_checkCertificateService(vc_serviceList)) {

            vc_oEMProvisioningCert := f_getCertificate(vc_IdCertificateMap, cc_crt_oem_leaf_valid);

            if(getverdict == pass) {
                vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
                vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.
                                                 tBSCertificate.serial_Number));

                var BodyType myCertificateInstallationBodyType := md_CMN_BodyType_001(
                                                md_SECC_CMN_CertificateInstallationReq_001(
                                                "ciReq",
                                                vc_oEMProvisioningCert,
                                                vc_x509IssuerName,
                                                vc_x509SerialNumber));

                var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                               cc_crt_oem_leaf_valid);

                var Signature aSignature := md_CMN_Signature_001(
                                            md_CMN_SignedInfoType_001("ciReq",
                                            f_generateDigestFromBodyType(
                                            myCertificateInstallationBodyType)),
                                            v_privateKey);

                tc_V2G_EVCC_Msg_Timer.start(
                par_V2G_EVCC_Msg_Timeout_CertificateInstallationReq);

                pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_003(
                                              vc_SessionID, aSignature),
                                              myCertificateInstallationBodyType));
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
alt {
[]pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                              md_CMN_MessageHeaderType_003(vc_SessionID,
                              md_CMN_Signature_002(
                              mw_CMN_SignedInfoType_003)),
                              md_CMN_BodyType_001(
                              mdw_SECC_CMN_CertificateInstallationRes_001(
                              m_CMN_sAProvisioningCertificateChain_001(),
                              m_CMN_contractSignatureCertChain_001()))))
                              -> value v_responseMessage {

    tc_V2G_EVCC_Msg_Timer.stop;

    vc_sAProvisioningCertChain := v_responseMessage.body.bodyElement.
                                     certificateInstallationRes.
                                     sAProvisioningCertificateChain;
    vc_contractSignatureCertChain := v_responseMessage.body.bodyElement.
                                   certificateInstallationRes.
                                   contractSignatureCertChain;
    vc_contractSignatureEncryptedPrivateKey := v_responseMessage.body.
                                     bodyElement.
                                     certificateInstallationRes.
                                     contractSignatureEncryptedPrivateKey;
    vc_dHpublickey := v_responseMessage.body.bodyElement.
                      certificateInstallationRes.dHpublickey;

    vc_eMAID := v_responseMessage.body.bodyElement.
                certificateInstallationRes.eMAID;

    if(getverdict == pass) {

        var Signature authSignature := v_responseMessage.header.signature_;

        var BodyType certificateInstallationResBodyType := v_responseMessage.
                                                           body;

        f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                    cc_algorithm_method_2,
                                    cc_algorithm_method_3, v_vct);

        if(getverdict == pass) {

                isSignatureOk := f_verifySignature(
                                 certificateInstallationResBodyType,
                                 authSignature,
                                 vc_sAProvisioningCertChain.
                                 x509Certificate, v_vct);

            if(isSignatureOk) {
                isCertChainOk := fx_verifyCertChain(vc_sAProvisioningCertChain,
                                                    vc_v2gRootCert, "CPS");
            } else {
               isCertChainOk := true;
            }
            if(not isCertChainOk) {
                    setverdict(v_vct, "CertChain of the oEMProvisioningCert " &
                                      "is not valid.");
            }
            if(isCertChainOk) {
                var boolean result := fx_checkCertificateValidity(
                                        vc_contractSignatureCertChain.
                                        x509Certificate.tBSCertificate.
                                        validity.not_Before,
                                        vc_contractSignatureCertChain.
                                        x509Certificate.tBSCertificate.
                                        validity.not_After);

                if(not result) {
                        setverdict(v_vct,"Invalid Certificate validity" &
                                         " was detected.");
                }
                if(getverdict == pass) {
                    log("EMAID: " & toString<charstring>(vc_eMAID.base));
                    result := fx_validateEMAID(vc_eMAID.base);
                    if(not result) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        setverdict(v_vct,"Invalid EMAID syntax was detected.");
                                    }
                                    else {
                                        setverdict(pass, "CertificateInstallationRes " &
                                                         "message was correct.");
                                    }

                                    vc_sharedSecret := fx_calculateSharedSecret(cc_crt_oem_leaf_valid,
                                                                                vc_dHpublickey);

                                    vc_privateKey := fx_decryptContractPrivateKey(
                                                     vc_sharedSecret,
                                                     vc_contractSignatureEncryptedPrivateKey);

                                    // check contract cert private key with given
                                    // NEW contract cert (public key)
                                      var boolean isValidContractPrivateKey :=
                                                              fx_isPrivateKeyValidForCertificate(
                                                            vc_privateKey,
                                                              vc_contractSignatureCertChain.
                                                              x509Certificate);

                                      if (not isValidContractPrivateKey) {
                                            setverdict(v_vct, "Contract private key does " &
                                                              "not match the " &
                                                              "contract certificate.");
                                      } else {

fx_storeContractCertificateChain(vc_contractSignatureCertChain,
                                                                                vc_privateKey);
                                        log("VALID contract private key to contract certificate: ",
                                            isValidContractPrivateKey);
                                      }
                                  }
                              }
                          }
                      }
                  }
                  [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                                 "was misleadingly " &
                                                                 "terminated by the SUT.");
                  [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                                   {certificateInstallationRes := ?});
                  [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                               "but cannot be decoded.");
                  [] a_SECC_Unexpected_Message_Content(v_vct, "Received CertificateInstallationRes " &
                                                              "message, but with invalid " &
                                                              "data content.");
                  [] a_SECC_Timeout(v_vct);
                  }
              }
          }
          else {setverdict(inconc,"Wrong PIXIT selection. No certificate service has been offered.");}
          return getverdict;
        }

      function f_SECC_CMN_TB_VTB_CertificateInstallation_002() runs on SECC_Tester
                                                               return verdicttype  {

          var charstring v_message;
          var boolean isShutdownOSC := false;
          var boolean isClosedTCP := false;
          var boolean isErrorMessage := false;
          var charstring v_charstring := "";

              f_SECC_setIsConfirmationFlagDC(true);
              f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                               vc_FrequencyOff);
              f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                               vc_DutyCycleOscOff);
              deactivate(vc_Default_IEC_61851_ListenerBehavior);

          tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
          tc_CMN_TCP_Connection_Termination_Timer.start(
          par_CMN_TCP_Connection_Termination_Timeout);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        []    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                          fAILED_SequenceError)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                            "'fAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                       e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                       {serviceDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateInstallationRes " &
                                                   "message, but with invalid " &
                                                   "data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                        "message with responseCode 'fAILED_SequenceError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
  }

function f_SECC_CMN_TB_VTB_CertificateInstallation_003() runs on SECC_Tester
                                                         return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
vc_oEMProvisioningCert :=
f_getCertificate(vc_IdCertificateMap, cc_crt_oem_leaf_valid);
if(getverdict == pass) {
    vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
    vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.
                                  tBSCertificate.serial_Number));

    var BodyType myCertificateInstallationBodyType := md_CMN_BodyType_001(
                                         md_SECC_CMN_CertificateInstallationReq_001(
                                         "ciReq",
                                         vc_oEMProvisioningCert,
                                         vc_x509IssuerName,
                                         vc_x509SerialNumber));

    var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                   cc_crt_oem_leaf_valid);

    var Signature aSignature := md_CMN_Signature_001(
                               md_CMN_SignedInfoType_001("ciReq",
                               f_generateDigestFromBodyType(
                               myCertificateInstallationBodyType)), v_privateKey);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateInstallationReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_003(
                                f_rnd_SessionID(1,429496), aSignature),
                                myCertificateInstallationBodyType));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_002(?),
                                         md_CMN_BodyType_001(
                                         mdw_SECC_CMN_CertificateInstallationRes_002(
                                         fAILED_UnknownSession)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"CertificateInstallationRes with responseCode " &
                            "'fAILED_UnknownSession' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                       e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                      {certificateInstallationRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateInstallationRes " &
                                                   "message, but with invalid " &
                                                   "data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a CertificateInstallationRes " &
                            "message with responseCode 'fAILED_UnknownSession' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    function f_SECC_CMN_TB_VTB_CertificateInstallation_004() runs on SECC_Tester
                                                             return verdicttype {

        timer t_wait;
        var boolean isShutdownOSC := false;
        var boolean isClosedTCP := false;
        var float v_timeShift := 1.0;

        // Wait until sequence timeout in SUT should expire
        t_wait.start(par_V2G_SECC_Sequence_Timeout -
                     v_timeShift);
        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                     e_getPortStatus));

        alt {
            [] t_wait.timeout;
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                          "terminated by the SUT " &
                                          "before timeout has expired.");
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
                setverdict(fail, "No V2G message was expected.");
            }
        }

        if(getverdict == pass) {

                f_SECC_setIsConfirmationFlagDC(true);
                f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                                 vc_FrequencyOff);
                f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                                 vc_DutyCycleOscOff);
                deactivate(vc_Default_IEC_61851_ListenerBehavior);

            sleep(v_timeShift - par_CMN_Transmission_Delay);

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                []
                a_Shutdown_Oscillator(pass){
                      isShutdownOSC := true;
                      if(isClosedTCP == false) {repeat;}
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                        "terminated by the SUT.") {
                     isClosedTCP := true;
                     tc_CMN_TCP_Connection_Termination_Timer.stop;
                     if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                                "TCP connection was terminated.");}
                     pt_HAL_61851_Internal_Port.clear;
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                         "but TCP connection was " &
                                                         ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout() {
                    setverdict(fail,"SUT has not terminated the TCP connection " &
                                    "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
function f_SECC_CMN_TB_VTB_CertificateInstallation_005() runs on SECC_Tester
                                                         return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    vc_oEMProvisioningCert :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_oem_leaf_valid);
    if(getverdict == pass) {
        vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
        vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.
                                       tBSCertificate.serial_Number));

        var BodyType myCertificateInstallationBodyType := md_CMN_BodyType_001(
                                            md_SECC_CMN_CertificateInstallationReq_001(
                                            "ciReq",
                                            vc_oEMProvisioningCert,
                                            vc_x509IssuerName,
                                            vc_x509SerialNumber));
        // load wrong private key
        var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                       cc_crt_contract_leaf_expires);

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("ciReq",
                                    f_generateDigestFromBodyType(
                                    myCertificateInstallationBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateInstallationReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_003(
                                      vc_SessionID, aSignature),
                                      myCertificateInstallationBodyType));

        alt {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_002(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            mdw_SECC_CMN_CertificateInstallationRes_002(
                                            fAILED_SignatureError)))) {

                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"CertificateInstallationRes with responseCode " &
                                "'fAILED_SignatureError' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                         e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        "TCP connection was terminated.");}
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {certificateInstallationRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                  "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateInstallationRes " &
                                                 "message, but with invalid " &
                                                 "data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a CertificateInstallationRes " &
                        "message with responseCode 'fAILED_SignatureError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_CertificateInstallation_006() runs on SECC_Tester
                                                          return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    vc_oEMProvisioningCert :=
    f_getCertificate(vc_IdCertificateMap, cc_crt_oem_leaf_valid);
    if(getverdict == pass) {
        vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
        vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.
                                       tBSCertificate.serial_Number));

        var BodyType myCertificateInstallationBodyType := md_CMN_BodyType_001(
                                              md_SECC_CMN_CertificateInstallationReq_001(
                                              "ciReq",
                                              vc_oEMProvisioningCert,
                                              vc_x509IssuerName,
                                              vc_x509SerialNumber));

        var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                       cc_crt_oem_leaf_valid);

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("ciReq",
                                    f_generateDigestFromBodyType(
                                    myCertificateInstallationBodyType)), v_privateKey);

        // change id after signature generation
        myCertificateInstallationBodyType.bodyElement.
        certificateInstallationReq.id := "contentChanged";

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateInstallationReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_MessageHeaderType_003(
                                        vc_SessionID, aSignature),
                                        myCertificateInstallationBodyType));

            alt {
                []
                a_Shutdown_Oscillator(pass){
                     isShutdownOSC := true;
                     if(isClosedTCP == false) {repeat;}
                }
                []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_002(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                mdw_SECC_CMN_CertificateInstallationRes_002(
                                                fAILED_SignatureError)))) {

                    tc_V2G_EVCC_Msg_Timer.stop;
                    setverdict(pass,"CertificateInstallationRes with responseCode " &
                                    "'fAILED_SignatureError' is successful.");

                    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                                e_getPortStatus));
                    isErrorMessage := true;
                    repeat;
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                             "terminated by the SUT.") {
                     isClosedTCP := true;
                     tc_CMN_TCP_Connection_Termination_Timer.stop;
                     if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                                "TCP connection was terminated.");}
                     pt_HAL_61851_Internal_Port.clear;
                }
                [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                                {certificateInstallationRes := ?});
                [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                             "but cannot be decoded.");
                [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateInstallationRes " &
                                                            "message, but with invalid " &
                                                            "data content.");
                [] a_SECC_Timeout(fail);
                [] a_SECC_TCP_Connection_Termination_Timeout() {
                    v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                    "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
                }
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a CertificateInstallationRes " &
                            "message with responseCode 'fAILED_SignatureError' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    function f_SECC_CMN_TB_VTB_CertificateInstallation_007(in charstring v_certName,
                                                            in template ResponseCodeType
                                                            v_responseCode)
                                                            runs on SECC_Tester
                                                            return verdicttype {

        var ResponseCodeType v_responseCodeReceived;
        var boolean isShutdownOSC := false;
        var boolean isClosedTCP := false;
        var boolean isErrorMessage := false;
        var charstring v_charstring := "";
        var V2G_Message v_responseMessage;

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                     vc_DutyCycleOscOff);
    deactivate(vc_Default_IEC_61851_ListenerBehavior);

vc_oEMProvisioningCert :=
f_getCertificate(vc_IdCertificateMap, v_certName);
if(getverdict == pass) {
    vc_x509IssuerName := fx_getIssuerName(vc_v2gRootCert.tBSCertificate.issuer);
    vc_x509SerialNumber := hex2int(str2hex(vc_v2gRootCert.
                                   tBSCertificate.serial_Number));

    var BodyType myCertificateInstallationBodyType := md_CMN_BodyType_001(
                                          md_SECC_CMN_CertificateInstallationReq_001(
                                          "ciReq",
                                          vc_oEMProvisioningCert,
                                          vc_x509IssuerName,
                                          vc_x509SerialNumber));

    var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                   v_certName);

    var Signature aSignature := md_CMN_Signature_001(
                                md_CMN_SignedInfoType_001("ciReq",
                                f_generateDigestFromBodyType(
                                myCertificateInstallationBodyType)), v_privateKey);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CertificateInstallationReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_003(
                                  vc_SessionID, aSignature),
                                  myCertificateInstallationBodyType));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        []pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_002(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_CertificateInstallationRes_002(
                                          v_responseCode))))
                                          -> value v_responseMessage {

            tc_V2G_EVCC_Msg_Timer.stop;
            v_responseCodeReceived := v_responseMessage.body.bodyElement.
                                  certificateInstallationRes.responseCode;
            setverdict(pass,"CertificateInstallationRes with responseCode '" &
                            toString<ResponseCodeType>(v_responseCodeReceived) &
                             "' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {certificateInstallationRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received CertificateInstallationRes " &
                                                   "message, but with invalid " &
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                        "data content.");
                [] a_SECC_Timeout(fail);
                [] a_SECC_TCP_Connection_Termination_Timeout() {
                    v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                    "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
                }
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a CertificateInstallationRes " &
                            "message with corresponding responseCode " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    function f_checkCertificateService(in ServiceListType vc_serviceList) return boolean {
        if(ispresent(vc_serviceList.service_list)) {
            for (var integer i:=0; i<sizeof(vc_serviceList.service_list); i:=i + 1) {
                if(vc_serviceList.service_list[i].serviceType.serviceCategory ==
                   contractCertificate) {
                    return true;
                }
            }
        }
        return false;
    }
}
```

I.2.2.11 SECC functions for SessionStop

```
module TestBehavior_SECC_SessionStop {

      import from Timer_15118_2 all;
      import from Timer_15118_3 all;
      import from Timer_15118 all;
      import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
      import from ComponentsAndPorts all;
      import from TestBehavior_SECC_CommonBehavior all;
      import from Templates_CMN_SessionStop all;
      import from Templates_CMN_ServiceDiscovery all;
      import from Templates_SECC_ServiceDiscovery all;
      import from Templates_CMN_V2GMessage all;
      import from urn_iso_15118_2_2013_MsgDef all;
      import from DataStructure_V2G_TCP_TLS_Port_Control all;
      import from Templates_CMN_V2GTcpTlsPortControl all;
      import from TTlibrary_Logging all;
      import from Pics_15118 all;
      import from Services_HAL_61851_1 all;
      import from TestBehavior_SECC_VsPlLnkStatus all;
      import from Services_SECC_PLCLinkStatus all;
      import from Timer_15118 all;
      import from Pixit_15118 all;
      import from LibFunctions_15118_2 all;
      import from LibFunctions_15118_3 all;
      import from DataStructure_HAL_61851 all;
      import from TestBehavior_SECC_CmSetKey all;

      function f_SECC_CMN_TB_VTB_SessionStop_001(ChargingSessionType chargingSession,
                                                 HAL_61851_Listener v_HAL_61851_Listener,
                                                 in verdicttype v_vct)
                                                  runs on SECC_Tester return verdicttype {

          var V2G_Message v_responseMessage;
          var charstring v_message;
          var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_internalResponseMessage;
          var boolean isClosed := false;
          var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var ChargingSessionType chargingSessionTerminate := terminate;
timer t_SECC_CPOscillator_Shutdown_Timer;
var hexstring v_Nid;
var hexstring v_Nmk;


    if(chargingSession == chargingSessionTerminate) {
      v_HAL_61851_Listener.stop;
      v_HAL_61851_Listener.start(f_SECC_HAL_61851_Listener(false, false,
                                                           e_StateA));
      f_SECC_changeValidFrequencyRange(vc_FrequencyOff,vc_FrequencyOff);
      f_SECC_changeValidDutyCycleRange(0.0,0.0);
      f_SECC_changeValidStateCondition(e_StateE);
      deactivate(vc_Default_IEC_61851_ListenerBehavior);
  }
  else {
      f_SECC_setIsConfirmationFlagDC(true);
      f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                       vc_FrequencyOff);
      f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                       vc_DutyCycleOscOff);
      deactivate(vc_Default_IEC_61851_ListenerBehavior);
  }
tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionStopReq);
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_CMN_CMN_SessionStopReq_001(chargingSession))));

alt {
    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_CMN_CMN_SessionStopRes_001(oK)))) {

      tc_V2G_EVCC_Msg_Timer.stop;
      pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                       e_disconnectPort));
      alt{
          [] pt_V2G_TCP_TLS_Port_Control.receive(md_CMN_CMN_tcpTlsStatusRes_001(
                                                 e_disconnectPort,?,?,*))
                                                 -> value v_internalResponseMessage {

               isClosed := v_internalResponseMessage.result.portStatus.closed;
               ackType := v_internalResponseMessage.ack;
          }
          []
               a_Shutdown_Oscillator(pass){
               vc_isShutdownOSC := true;
               repeat;
          }
      }


      if(isClosed and (ackType == ACK_SUCCESS)){
           vc_isTcpPortClosed := isClosed;
           setverdict(pass,"SessionStopRes message was correct.");
      } else {
           setverdict(v_vct,"TCP socket could not be closed.");
      }
      if(PICS_CMN_CMN_CombinedTesting and
         chargingSession == chargingSessionTerminate) {
          if(vc_unmapSlac) {
              map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
          }
          f_SECC_getPLCLinkTermination(
          par_T_match_leave + par_CMN_Transmission_Delay, v_vct);
          if(vc_unmapSlac) {
             unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
          }
      } else if(PICS_CMN_CMN_CombinedTesting and
                chargingSession == pause) {

          if(not vc_isShutdownOSC) {
              t_SECC_CPOscillator_Shutdown_Timer.start(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    par_SECC_CPOscillator_Shutdown_Timeout);
                    alt{
                        [] a_Shutdown_Oscillator(pass);
                        [] t_SECC_CPOscillator_Shutdown_Timer.timeout {
                            setverdict(fail, "SUT has not executed an " &
                                             "oscillator shutdown within " &
                                             "'par_SECC_CPOscillator_Shutdown_Timeout' " &
                                             "after a pause was initiated by the EV.");
                        }
                    }
                }
                if(getverdict == pass) {
                    setverdict(pass, "SUT has executed an " &
                                     "oscillator shutdown within " &
                                     "'par_SECC_CPOscillator_Shutdown_Timeout' " &
                                     "after a pause was initiated by the EV.");
                    v_HAL_61851_Listener.stop;
                    v_HAL_61851_Listener.start(f_SECC_HAL_61851_Listener(false, false,
                                                                         e_StateA));
                    if(PICS_CMN_CMN_WakeUp < par_SECC_T_step_X1) {
                        tc_T_step_X1.start(par_SECC_T_step_X1);
                    }
                }
            }
           }
        [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                       "was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {sessionStopRes := ?});
        [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(v_vct, "Received SessionStopRes message, " &
                                                    "but with invalid data content.");
        [] a_SECC_Timeout(v_vct);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SessionStop_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                         fAILED_SequenceError)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                            "'fAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                     {serviceDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received SessionStopRes message, " &
                                                    "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                        "message with responseCode 'fAILED_SequenceError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SessionStop_003(ChargingSessionType chargingSession)
                                         runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                     vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                     vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionStopReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(
                                 f_rnd_SessionID(1,429496)),md_CMN_BodyType_001(
                                 md_CMN_CMN_SessionStopReq_001(chargingSession))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(?),
                                         md_CMN_BodyType_001(
                                         md_CMN_CMN_SessionStopRes_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        fAILED_UnknownSession)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"SessionStopRes with responseCode " &
                            "'fAILED_UnknownSession' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {sessionStopRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received SessionStopRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a SessionStopRes " &
                        "message with responseCode 'fAILED_UnknownSession' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SessionStop_004() runs on SECC_Tester return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var float v_timeShift := 1.0;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
                 v_timeShift);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                        "terminated by the SUT " &
                                        "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                        vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                        vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        LibFunctions_15118_2.sleep(v_timeShift - par_CMN_Transmission_Delay);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_SECC_CMN_TB_VTB_SessionStop_005(ChargingSessionType chargingSession)
                                          runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_SessionStopReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_SessionStopReq_001(chargingSession))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_SessionStopRes_001(fAILED)))) {

            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"SessionStopRes with responseCode 'fAILED' is successful. " &
                            "SUT could not measuring state B before.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {sessionStopRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received SessionStopRes message, " &
                                                        "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout' " &
                                "or other message data content was not conform. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a SessionStopRes " &
                            "message with responseCode 'fAILED'. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }
}
```

I.2.2.12 SECC functions for ChargingStatus

```
module TestBehavior_SECC_ChargingStatus {

    import from Pixit_15118 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Pixit_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from Timer_15118_2 all;
    import from ComponentsAndPorts all;
    import from Templates_CMN_ChargingStatus all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from Pics_15118_2 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118 all;
    import from DataStructure_HAL_61851 all;

    function f_SECC_AC_TB_VTB_ChargingStatus_001(in verdicttype v_vct)
                                                 runs on SECC_Tester
                                                 return verdicttype {

         var V2G_Message v_responseMessage;
         var charstring v_message;

         tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargingStatusReq);

         pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(m_CMN_AC_ChargingStatusReq_001)));
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      alt{
         [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001 (
                                          md_CMN_AC_ChargingStatusRes_001 (
                                          oK, vc_EvseId, vc_SAScheduleTupleId,*,*,*,
                                          md_CMN_AC_EVSEStatus_001(?,?,?)))))
                                          -> value v_responseMessage {

          tc_V2G_EVCC_Msg_Timer.stop;
          vc_EvseId := v_responseMessage.body.bodyElement.chargingStatusRes.eVSEID;
          if(vc_EvseId != cc_EvseId_NotSup) {
              var boolean result := fx_validateEVSEID(vc_EvseId, "ISO");
              if(not result) {
                      setverdict(v_vct,"Invalid EVSEID syntax was detected.");
              }
          }
          if(ispresent(v_responseMessage.body.bodyElement.
                       chargingStatusRes.receiptRequired)) {
              vc_receiptRequired := v_responseMessage.body.bodyElement.
                                    chargingStatusRes.receiptRequired;
              if(PICS_CMN_CMN_IdentificationMode == eIM) {
                  if(vc_receiptRequired == true){
                         setverdict(v_vct, "ReceiptRequired shall always set to false " &
                                           "in EIM Identification Mode.");
                  }
              }
              else {
                 if(PIXIT_SECC_CMN_ReceiptRequired == receiptRequired) {
                     if(vc_receiptRequired == false){
                             setverdict(v_vct, "ReceiptRequired shall always set to true " &
                                               "if the SECC requires a metering receipt " &
                                               "from the EVCC.");
                     }
                 }
                 else if(PIXIT_SECC_CMN_ReceiptRequired == none_){
                     if(vc_receiptRequired == true){
                             setverdict(v_vct, "ReceiptRequired shall always set to false " &
                                               "if the SECC not requires a metering receipt " &
                                               "from the EVCC.");
                     }
                 }
              }
          }
          if(getverdict == pass) {
              setverdict(pass, "ChargingStatusRes message was correct.");
              vc_EVSENotification := v_responseMessage.body.bodyElement.
                                     chargingStatusRes.aC_EVSEStatus.eVSENotification;
              if(ispresent(v_responseMessage.body.bodyElement.chargingStatusRes.
                           meterInfo)) {
                  vc_MeterInfo := v_responseMessage.body.bodyElement.
                                  chargingStatusRes.meterInfo;
              }
          }
         }
         [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                        "terminated by the SUT.");
         [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                         {chargingStatusRes := ?});
         [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                      "but cannot be decoded.");
         [] a_SECC_Unexpected_Message_Content(v_vct, "Received ChargingStatusRes message, " &
                                                     "but with invalid data content.");
         [] a_SECC_Timeout(v_vct);
       }
  return getverdict();
 }

 function f_SECC_AC_TB_VTB_ChargingStatus_002() runs on SECC_Tester return verdicttype {

     var charstring v_message;
     var boolean isShutdownOSC := false;
     var boolean isClosedTCP := false;
     var boolean isErrorMessage := false;
     var charstring v_charstring := "";
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryReq_002(omit))));
        alt{
           []
           a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
           }
           [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                                fAILED_SequenceError)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                               "'fAILED_SequenceError' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
           }
           [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT.") {
                isClosedTCP := true;
                tc_CMN_TCP_Connection_Termination_Timer.stop;
                if(isShutdownOSC == false) {
                   logWarn("SECC oscillator was not shut down before the " &
                           "TCP connection was terminated.");}
                pt_HAL_61851_Internal_Port.clear;
           }
           [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {serviceDiscoveryRes := ?});
           [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                       "but cannot be decoded.");
           [] a_SECC_Unexpected_Message_Content(fail, "Received ChargingStatusRes message, " &
                                                         "but with invalid data content.");
           [] a_SECC_Timeout(fail);
           [] a_SECC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
           }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                        "message with responseCode 'fAILED_SequenceError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict();
   }

   function f_SECC_AC_TB_VTB_ChargingStatus_003() runs on SECC_Tester return verdicttype {

       var charstring v_message;
       var boolean isShutdownOSC := false;
       var boolean isClosedTCP := false;
       var boolean isErrorMessage := false;
       var charstring v_charstring := "";
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

     tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargingStatusReq);
     tc_CMN_TCP_Connection_Termination_Timer.start(
     par_CMN_TCP_Connection_Termination_Timeout);

     pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    f_rnd_SessionID(1,429496)),
                                    md_CMN_BodyType_001(
                                    m_CMN_AC_ChargingStatusReq_001)));
     alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(?),
                                              md_CMN_BodyType_001 (
                                            md_CMN_AC_ChargingStatusRes_001(
                                            fAILED_UnknownSession,?,?,
                                            omit,omit,omit,?)))) {
             tc_V2G_EVCC_Msg_Timer.stop;
             setverdict(pass,"ChargingStatusRes with responseCode " &
                              "'fAILED_UnknownSession' is successful.");

             pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                              e_getPortStatus));
             isErrorMessage := true;
             repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                         {chargingStatusRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received ChargingStatusRes message, " &
                                                     "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ChargingStatusRes " &
                        "message with responseCode 'fAILED_UnknownSession' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict();
}

function f_SECC_AC_TB_VTB_ChargingStatus_004() runs on SECC_Tester return verdicttype {

  timer t_wait;
  var boolean isShutdownOSC := false;
  var boolean isClosedTCP := false;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  var float v_timeShift := 1.0;

  // Wait until sequence timeout in SUT should expire
  t_wait.start(par_V2G_SECC_Sequence_Timeout -
               v_timeShift);
  pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

  alt {
      [] t_wait.timeout;
      [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                    "terminated by the SUT " &
                                                    "before timeout has expired.");
      [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
          setverdict(fail, "No V2G message was expected.");
      }
  }

  if(getverdict == pass) {

          f_SECC_setIsConfirmationFlagDC(true);
          f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                           vc_FrequencyOff);
          f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                           vc_DutyCycleOscOff);
          deactivate(vc_Default_IEC_61851_ListenerBehavior);

      sleep(v_timeShift - par_CMN_Transmission_Delay);

      tc_CMN_TCP_Connection_Termination_Timer.start(
      par_CMN_TCP_Connection_Termination_Timeout);

      alt {
          []
          a_Shutdown_Oscillator(pass){
                isShutdownOSC := true;
                if(isClosedTCP == false) {repeat;}
          }
          [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                        "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if(isShutdownOSC == false) {
                  logWarn("SECC oscillator was not shut down before the " &
                          "TCP connection was terminated.");}
               pt_HAL_61851_Internal_Port.clear;
          }
          [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                     "but TCP connection was " &
                                                     ">>NOT<< terminated by the SUT");
          [] a_SECC_TCP_Connection_Termination_Timeout() {
              setverdict(fail,"SUT has not terminated the TCP connection " &
                              "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
          }
      }
  }
  return getverdict;
}

function f_SECC_AC_TB_VTB_ChargingStatus_005() runs on SECC_Tester return verdicttype {

     var V2G_Message v_responseMessage;
     var charstring v_message;

     tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargingStatusReq);

     pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(m_CMN_AC_ChargingStatusReq_001)));
     alt
      {
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001 (
                                          md_CMN_AC_ChargingStatusRes_001 (
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        oK, vc_EvseId, vc_SAScheduleTupleId,*,?,*,
                                        md_CMN_AC_EVSEStatus_001(?,?,?)))))
                                        -> value v_responseMessage {

            tc_V2G_EVCC_Msg_Timer.stop;
            vc_EvseId := v_responseMessage.body.bodyElement.chargingStatusRes.eVSEID;
            vc_MeterInfo := v_responseMessage.body.bodyElement.chargingStatusRes.
                            meterInfo;
            if(vc_EvseId != cc_EvseId_NotSup) {
                var boolean result := fx_validateEVSEID(vc_EvseId, "ISO");
                if(not result) {
                    setverdict(fail,"Invalid EVSEID syntax was detected.");
                }
            }
            if(ispresent(v_responseMessage.body.bodyElement.
                         chargingStatusRes.receiptRequired)) {
                vc_receiptRequired := v_responseMessage.body.bodyElement.
                                      chargingStatusRes.receiptRequired;
                if(PICS_CMN_CMN_IdentificationMode == eIM) {
                    if(vc_receiptRequired == true){
                        setverdict(fail, "ReceiptRequired shall always set to false " &
                                         "in EIM Identification Mode.");
                    }
                }
                else {
                   if(PIXIT_SECC_CMN_ReceiptRequired == receiptRequired) {
                       if(vc_receiptRequired == false){
                           setverdict(fail, "ReceiptRequired shall always set to true " &
                                            "if the SECC requires a metering receipt " &
                                            "from the EVCC");
                       }
                   }
                   else if(PIXIT_SECC_CMN_ReceiptRequired == none_){
                       if(vc_receiptRequired == true){
                           setverdict(fail, "ReceiptRequired shall always set to false " &
                                            "if the SECC not requires a metering receipt " &
                                            "from the EVCC");
                       }
                   }
                }
            }
            if(ispresent(vc_MeterInfo.meterReading)) {
                 if(vc_MeterInfo.meterReading < 0 or
                    vc_MeterInfo.meterReading > cc_maximum_value_meterReading) {
                     setverdict(fail, "Invalid value range for MeterReading detected.");
                 }
            }

            if(getverdict == pass) {
                setverdict(pass, "ChargingStatusRes message was correct.");
                vc_EVSENotification := v_responseMessage.body.bodyElement.
                                       chargingStatusRes.aC_EVSEStatus.eVSENotification;
            }
           }
           [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                         "terminated by the SUT.");
           [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {chargingStatusRes := ?});
           [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                       "but cannot be decoded.");
           [] a_SECC_Unexpected_Message_Content(fail, "Received ChargingStatusRes message, " &
                                                      "but with invalid data content.");
           [] a_SECC_Timeout(fail);
         }
   return getverdict();
  }
}
```

I.2.2.13 SECC functions for MeteringReceipt

```
module TestBehavior_SECC_MeteringReceipt {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Timer_15118_2 all;
import from Pics_15118_2 all;
import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
import from ComponentsAndPorts all;
import from Templates_SECC_MeteringReceipt all;
import from Templates_CMN_ServiceDiscovery all;
import from Templates_SECC_ServiceDiscovery all;
import from Templates_CMN_V2GMessage all;
import from TestBehavior_SECC_CommonBehavior all;
import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
import from TTlibrary_Logging all;
import from Pics_15118 all;
import from Services_HAL_61851_1 all;
import from Templates_CMN_PowerDelivery all;
import from Templates_CMN_ChargingStatus all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
import from Services_Security all;
import from Templates_CMN_Security all;
import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
import from LibFunctions_15118_2 all;
import from urn_iso_15118_2_2013_X509Def language "XSD" all;
import from Timer_15118 all;

function f_SECC_AC_TB_VTB_MeteringReceipt_001(in verdicttype v_vct)
                                              runs on SECC_Tester
                                              return verdicttype {

    var V2G_Message v_responseMessage;
    var charstring v_message;
    var charstring v_privateKey;

    if(vc_receiptRequired == true){

        var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                                  md_SECC_CMN_MeteringReceiptReq_001(
                                                  "mReq", vc_SessionID,
                                                  vc_MeterInfo, vc_SAScheduleTupleId));

        if(isbound(vc_privateKey)) {
            v_privateKey := vc_privateKey;
        } else {
            v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
        }

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("mReq",
                                    f_generateDigestFromBodyType(myMeteringReceiptBodyType)),
                                    v_privateKey);



        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_003(vc_SessionID, aSignature),
                                      myMeteringReceiptBodyType));

        alt {
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001 (
                                            md_SECC_AC_MeteringReceiptRes_001(oK,
                                            md_CMN_AC_EVSEStatus_001(?,?,?)))))
                                            -> value v_responseMessage {
               tc_V2G_EVCC_Msg_Timer.stop;
               setverdict(pass, "MeteringReceiptRes message was correct.");
            }
            [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                           "was misleadingly " &
                                                           "terminated by the SUT.");
            [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {meteringReceiptRes := ?});
            [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_SECC_Unexpected_Message_Content(v_vct, "Received MeteringReceiptRes message, " &
                                                        "but with invalid data content.");
            [] a_SECC_Timeout(v_vct);
        }
    } else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
                              "set to 'true' by the SUT before.");}
    return getverdict;
}

function f_SECC_AC_TB_VTB_MeteringReceipt_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var charstring v_privateKey;

    if(vc_receiptRequired == true){

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                                   md_SECC_CMN_MeteringReceiptReq_001(
                                                   "mReq", vc_SessionID,
                                                   vc_MeterInfo, vc_SAScheduleTupleId));

        if(isbound(vc_privateKey)) {
            v_privateKey := vc_privateKey;
        } else {
            v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
        }

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("mReq",
                                    f_generateDigestFromBodyType(myMeteringReceiptBodyType)),
                                    v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_003(
                                          f_rnd_SessionID(1,429496), aSignature),
                                          myMeteringReceiptBodyType));

        alt {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                    md_CMN_MessageHeaderType_001(?),
                                                    md_CMN_BodyType_001 (
                                                    md_SECC_AC_MeteringReceiptRes_001(
                                                    fAILED_UnknownSession,?)))){
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"MeteringReceiptRes with responseCode " &
                                "'fAILED_UnknownSession' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                                 e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                tc_CMN_TCP_Connection_Termination_Timer.stop;
                if(isShutdownOSC == false) {
                   logWarn("SECC oscillator was not shut down before the " &
                           "TCP connection was terminated.");}
                pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                             {meteringReceiptRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                   "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received MeteringReceiptRes message, " &
                                                  "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a MeteringReceiptRes " &
                            "message with responseCode 'fAILED_UnknownSession' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    } else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
                      "set to 'true' by the SUT before.");}
    return getverdict;
}

function f_SECC_AC_TB_VTB_MeteringReceipt_003() runs on SECC_Tester return verdicttype {

    var V2G_Message v_responseMessage;
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var charstring v_privateKey;

    if(vc_receiptRequired == true){

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        // set invalid meter reading
        var integer v_meterReading := vc_MeterInfo.meterReading;
        v_meterReading := v_meterReading + 1000;
        vc_MeterInfo.meterReading := v_meterReading;

        var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                                   md_SECC_CMN_MeteringReceiptReq_001(
                                                   "mReq", vc_SessionID,
                                                   vc_MeterInfo, vc_SAScheduleTupleId));

        if(isbound(vc_privateKey)) {
            v_privateKey := vc_privateKey;
        } else {
            v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
        }

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("mReq",
                                    f_generateDigestFromBodyType(myMeteringReceiptBodyType)),
                                    v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_003(vc_SessionID, aSignature),
                                myMeteringReceiptBodyType));

        alt {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001 (
                                          md_SECC_AC_MeteringReceiptRes_001(
                                          fAILED_MeteringSignatureNotValid,?)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"MeteringReceiptRes with responseCode " &
                                "'fAILED_MeteringSignatureNotValid' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                             e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {meteringReceiptRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received MeteringReceiptRes message, " &
                                                       "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a MeteringReceiptRes " &
                            "message with responseCode 'fAILED_MeteringSignatureNotValid' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    } else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
                      "set to 'true' by the SUT before.");}
    return getverdict;
}

function f_SECC_AC_TB_VTB_MeteringReceipt_004() runs on SECC_Tester return verdicttype {

    var V2G_Message v_responseMessage;
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

    if(vc_receiptRequired == true){

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                              md_SECC_CMN_MeteringReceiptReq_001(
                                              "mReq", vc_SessionID,
                                              vc_MeterInfo, vc_SAScheduleTupleId));

    // load wrong private key
    var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                   cc_crt_oem_leaf_valid);

    var Signature aSignature := md_CMN_Signature_001(
                                md_CMN_SignedInfoType_001("mReq",
                                f_generateDigestFromBodyType(myMeteringReceiptBodyType)),
                                v_privateKey);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_003(vc_SessionID, aSignature),
                                  myMeteringReceiptBodyType));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001 (
                                          md_SECC_AC_MeteringReceiptRes_001(
                                          fAILED_MeteringSignatureNotValid,?)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"MeteringReceiptRes with responseCode " &
                            "'fAILED_MeteringSignatureNotValid' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {meteringReceiptRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received MeteringReceiptRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a MeteringReceiptRes " &
                        "message with responseCode 'fAILED_MeteringSignatureNotValid' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
} else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
          "set to 'true' by the SUT before.");}
return getverdict;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}

function f_SECC_AC_TB_VTB_MeteringReceipt_005() runs on SECC_Tester return verdicttype {

    var V2G_Message v_responseMessage;
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var charstring v_privateKey;

    if(vc_receiptRequired == true){

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                                   md_SECC_CMN_MeteringReceiptReq_001(
                                                   "mReq", vc_SessionID,
                                                   vc_MeterInfo, vc_SAScheduleTupleId));

        if(isbound(vc_privateKey)) {
            v_privateKey := vc_privateKey;
        } else {
            v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
        }

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("mReq",
                                    f_generateDigestFromBodyType(myMeteringReceiptBodyType)),
                                    v_privateKey);

        // change meter reading after signature generation
        var integer v_meterReading := vc_MeterInfo.meterReading;
        v_meterReading := v_meterReading + 1000;
        vc_MeterInfo.meterReading := v_meterReading;
        myMeteringReceiptBodyType.bodyElement.meteringReceiptReq.meterInfo := vc_MeterInfo;

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_003(vc_SessionID, aSignature),
                                    myMeteringReceiptBodyType));

        alt {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001 (
                                            md_SECC_AC_MeteringReceiptRes_001(
                                            fAILED_MeteringSignatureNotValid,?)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"MeteringReceiptRes with responseCode " &
                                "'fAILED_MeteringSignatureNotValid' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    if(isShutdownOSC == false) {
                       logWarn("SECC oscillator was not shut down before the " &
                               "TCP connection was terminated.");}
                    pt_HAL_61851_Internal_Port.clear;
                }
                [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                                 {meteringReceiptRes := ?});
                [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
                [] a_SECC_Unexpected_Message_Content(fail, "Received MeteringReceiptRes message, " &
                                                       "but with invalid data content.");
                [] a_SECC_Timeout(fail);
                [] a_SECC_TCP_Connection_Termination_Timeout() {
                    v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                    "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
                }
            }
            if(not isErrorMessage) {
                v_charstring := v_charstring & "SUT has not sent a MeteringReceiptRes " &
                                "message with responseCode 'fAILED_MeteringSignatureNotValid' " &
                                "or other message data content was not conform. ";
            }
            if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
               setverdict(fail,v_charstring);
            }
        } else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
                  "set to 'true' by the SUT before.");}
        return getverdict;
    }

    function f_SECC_DC_TB_VTB_MeteringReceipt_001(in verdicttype v_vct)
                                                  runs on SECC_Tester
                                                  return verdicttype {

        var V2G_Message v_responseMessage;
        var charstring v_message;
        var charstring v_privateKey;

        if(vc_receiptRequired == true){

            var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                                      md_SECC_CMN_MeteringReceiptReq_001(
                                                      "mReq", vc_SessionID,
                                                      vc_MeterInfo, vc_SAScheduleTupleId));

            if(isbound(vc_privateKey)) {
                v_privateKey := vc_privateKey;
            } else {
                v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
            }

            var Signature aSignature := md_CMN_Signature_001(
                                        md_CMN_SignedInfoType_001("mReq",
                                        f_generateDigestFromBodyType(myMeteringReceiptBodyType)),
                                        v_privateKey);

            tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_003(vc_SessionID, aSignature),
                                          myMeteringReceiptBodyType));

            alt {
                [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001 (
                                                 md_SECC_DC_MeteringReceiptRes_001(
                                                 oK,md_CMN_DC_EVSEStatus_001(?,?,*,?)))))
                                                 -> value v_responseMessage {
                   tc_V2G_EVCC_Msg_Timer.stop;
                   if(ispresent(v_responseMessage.body.bodyElement.meteringReceiptRes.eVSEStatus.
                                dC_EVSEStatus.eVSEIsolationStatus)) {
                       if(v_responseMessage.body.bodyElement.meteringReceiptRes.eVSEStatus.
                             dC_EVSEStatus.eVSEIsolationStatus != valid) {
                          logWarn("The field 'eVSEIsolationStatus' should be set to 'valid'.");
                       }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                }
                if(v_responseMessage.body.bodyElement.meteringReceiptRes.eVSEStatus.
                      dC_EVSEStatus.eVSEStatusCode != eVSE_Ready) {
                   logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
                }
                setverdict(pass, "MeteringReceiptRes message was correct.");
            }
            [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                           "was misleadingly " &
                                                           "terminated by the SUT.");
            [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {meteringReceiptRes := ?});
            [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(v_vct, "Received MeteringReceiptRes message, " &
                                                        "but with invalid data content.");
            [] a_SECC_Timeout(v_vct);
        }
    } else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
              "set to 'true' by the SUT before.");}
    return getverdict;
}

function f_SECC_DC_TB_VTB_MeteringReceipt_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var charstring v_privateKey;

    if(vc_receiptRequired == true){

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                                   md_SECC_CMN_MeteringReceiptReq_001(
                                                   "mReq", vc_SessionID,
                                                   vc_MeterInfo, vc_SAScheduleTupleId));

        if(isbound(vc_privateKey)) {
            v_privateKey := vc_privateKey;
        } else {
            v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
        }

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001("mReq",
                                    f_generateDigestFromBodyType(
                                    myMeteringReceiptBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_003(
                                          f_rnd_SessionID(1,429496), aSignature),
                                          myMeteringReceiptBodyType));

        alt {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                    md_CMN_MessageHeaderType_001(?),
                                                    md_CMN_BodyType_001 (
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                md_SECC_DC_MeteringReceiptRes_001(
                                                fAILED_UnknownSession,?)))){
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"MeteringReceiptRes with responseCode " &
                                "'fAILED_UnknownSession' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {meteringReceiptRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received MeteringReceiptRes message, " &
                                                       "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a MeteringReceiptRes " &
                            "message with responseCode 'fAILED_UnknownSession' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    } else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
              "set to 'true' by the SUT before.");}
    return getverdict;
}

function f_SECC_DC_TB_VTB_MeteringReceipt_003() runs on SECC_Tester return verdicttype {

    var V2G_Message v_responseMessage;
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var charstring v_privateKey;

    if(vc_receiptRequired == true){

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        // set invalid meter reading
        var integer v_meterReading := vc_MeterInfo.meterReading;
        v_meterReading := v_meterReading + 1000;
        vc_MeterInfo.meterReading := v_meterReading;

        var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                                   md_SECC_CMN_MeteringReceiptReq_001(
                                                   "mReq", vc_SessionID,
                                                   vc_MeterInfo, vc_SAScheduleTupleId));

        if(isbound(vc_privateKey)) {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            v_privateKey := vc_privateKey;
        } else {
            v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
        }

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001(
                                    "mReq",
                                    f_generateDigestFromBodyType(
                                    myMeteringReceiptBodyType)), v_privateKey);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_003(vc_SessionID, aSignature),
                                      myMeteringReceiptBodyType));

        alt {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001 (
                                                 md_SECC_DC_MeteringReceiptRes_001(
                                                 fAILED_MeteringSignatureNotValid,?)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"MeteringReceiptRes with responseCode " &
                                "'fAILED_MeteringSignatureNotValid' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                             e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                             {meteringReceiptRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received MeteringReceiptRes message, " &
                                                       "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a MeteringReceiptRes " &
                            "message with responseCode 'fAILED_MeteringSignatureNotValid' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    } else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
              "set to 'true' by the SUT before.");}
    return getverdict;
}

function f_SECC_DC_TB_VTB_MeteringReceipt_004() runs on SECC_Tester return verdicttype {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var V2G_Message v_responseMessage;
var charstring v_message;
var boolean isShutdownOSC := false;
var boolean isClosedTCP := false;
var boolean isErrorMessage := false;
var charstring v_charstring := "";

if(vc_receiptRequired == true){

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                               md_SECC_CMN_MeteringReceiptReq_001(
                                               "mReq", vc_SessionID,
                                               vc_MeterInfo, vc_SAScheduleTupleId));

    // load wrong private key
    var charstring v_privateKey := f_getPrivateKey(vc_IdPrivateKeyMap,
                                                   cc_crt_oem_leaf_valid);

    var Signature aSignature := md_CMN_Signature_001(
                                md_CMN_SignedInfoType_001(
                                "mReq",
                                f_generateDigestFromBodyType(
                                myMeteringReceiptBodyType)), v_privateKey);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_003(vc_SessionID, aSignature),
                                  myMeteringReceiptBodyType));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001 (
                                            md_SECC_DC_MeteringReceiptRes_001(
                                            fAILED_MeteringSignatureNotValid,?)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"MeteringReceiptRes with responseCode " &
                            "'fAILED_MeteringSignatureNotValid' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                         e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                         {meteringReceiptRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received MeteringReceiptRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a MeteringReceiptRes " &
                            "message with responseCode 'fAILED_MeteringSignatureNotValid' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    } else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
              "set to 'true' by the SUT before.");}
    return getverdict;
}

function f_SECC_DC_TB_VTB_MeteringReceipt_005() runs on SECC_Tester return verdicttype {

    var V2G_Message v_responseMessage;
    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";
    var charstring v_privateKey;

    if(vc_receiptRequired == true){

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                                   md_SECC_CMN_MeteringReceiptReq_001(
                                                   "mReq", vc_SessionID,
                                                   vc_MeterInfo, vc_SAScheduleTupleId));

        if(isbound(vc_privateKey)) {
            v_privateKey := vc_privateKey;
        } else {
            v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
        }

        var Signature aSignature := md_CMN_Signature_001(
                                    md_CMN_SignedInfoType_001(
                                    "mReq",
                                    f_generateDigestFromBodyType(
                                    myMeteringReceiptBodyType)), v_privateKey);

        // change meter reading after signature generation
        var integer v_meterReading := vc_MeterInfo.meterReading;
        v_meterReading := v_meterReading + 1000;
        vc_MeterInfo.meterReading := v_meterReading;
        myMeteringReceiptBodyType.bodyElement.meteringReceiptReq.meterInfo := vc_MeterInfo;

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_003(vc_SessionID, aSignature),
                                    myMeteringReceiptBodyType));

        alt {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                md_CMN_BodyType_001 (
                                                md_SECC_DC_MeteringReceiptRes_001(
                                                fAILED_MeteringSignatureNotValid,?)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"MeteringReceiptRes with responseCode " &
                                "'fAILED_MeteringSignatureNotValid' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                            e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {meteringReceiptRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received MeteringReceiptRes message, " &
                                                       "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a MeteringReceiptRes " &
                            "message with responseCode 'fAILED_MeteringSignatureNotValid' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    } else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
              "set to 'true' by the SUT before.");}
    return getverdict;
}

function f_SECC_CMN_TB_VTB_MeteringReceipt_001() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

    if(vc_receiptRequired == true){

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

        alt {
            []
            a_Shutdown_Oscillator(pass){
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
             }
             []    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                                fAILED_SequenceError)))) {
                 tc_V2G_EVCC_Msg_Timer.stop;
                 setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                                "'fAILED_SequenceError' is successful.");

                 pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                             e_getPortStatus));
                 isErrorMessage := true;
                 repeat;
             }
             [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
             }
             [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {serviceDiscoveryRes := ?});
             [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                       "but cannot be decoded.");
             [] a_SECC_Unexpected_Message_Content(fail, "Received MeteringReceiptRes message, " &
                                                      "but with invalid data content.");
             [] a_SECC_Timeout(fail);
             [] a_SECC_TCP_Connection_Termination_Timeout() {
                 v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
             }
         }
         if(not isErrorMessage) {
             v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                            "message with responseCode 'fAILED_SequenceError' " &
                            "or other message data content was not conform. ";
         }
         if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
            setverdict(fail,v_charstring);
         }
     } else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
                "set to 'true' by the SUT before.");}
     return getverdict;
 }

 function f_SECC_CMN_TB_VTB_MeteringReceipt_002() runs on SECC_Tester return verdicttype {

     timer t_wait;
     var boolean isShutdownOSC := false;
     var boolean isClosedTCP := false;
     var float v_timeShift := 1.0;

     if(vc_receiptRequired == true){

         // Wait until sequence timeout in SUT should expire
         t_wait.start(par_V2G_SECC_Sequence_Timeout -
                      v_timeShift);
         pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));

         alt {
             [] t_wait.timeout;
             [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                            "terminated by the SUT " &
                                            "before timeout has expired.");
             [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
                 setverdict(fail, "No V2G message was expected.");
             }
         }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if(getverdict == pass) {

                    f_SECC_setIsConfirmationFlagDC(true);
                    f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                                     vc_FrequencyOff);
                    f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                                     vc_DutyCycleOscOff);
                    deactivate(vc_Default_IEC_61851_ListenerBehavior);

                sleep(v_timeShift - par_CMN_Transmission_Delay);

                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);

                alt {
                    []
                    a_Shutdown_Oscillator(pass){
                          isShutdownOSC := true;
                          if(isClosedTCP == false) {repeat;}
                    }
                    [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                                  "terminated by the SUT.") {
                         isClosedTCP := true;
                         tc_CMN_TCP_Connection_Termination_Timer.stop;
                         if(isShutdownOSC == false) {
                            logWarn("SECC oscillator was not shut down before the " &
                                    "TCP connection was terminated.");}
                         pt_HAL_61851_Internal_Port.clear;
                    }
                    [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                               "but TCP connection was " &
                                                               ">>NOT<< terminated by the SUT");
                    [] a_SECC_TCP_Connection_Termination_Timeout() {
                        setverdict(fail,"SUT has not terminated the TCP connection " &
                                        "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                    }
                }
            }
        } else {setverdict(inconc,"Wrong PIXIT selection. ReceiptRequired was not " &
                  "set to 'true' by the SUT before.");}
        return getverdict;
    }
}
```

I.2.2.14 SECC functions for CableCheck

```
module TestBehavior_SECC_CableCheck {

    import from Pics_15118 all;
    import from Pics_15118_2 all;
    import from Timer_15118_2 all;
    import from ComponentsAndPorts all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from Templates_CMN_CableCheck all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDef all;
       import from Configurations_15118_2 all;
    import from Timer_15118 all;
    import from DataStructure_HAL_61851 all;
    import from Services_HAL_61851_1 all;
    import from TTlibrary_Logging all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Pixit_15118_2 all;
    import from Pixit_15118 all;
    import from LibFunctions_15118_2 all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
function f_SECC_DC_TB_VTB_CableCheck_001(in HAL_61851_Listener v_HAL_61851_Listener,
                                         in verdicttype v_vct)
                                         runs on SECC_Tester return verdicttype  {

      var V2G_Message v_responseMessage;
      var charstring v_message;
      var boolean ongoingProcess := false;
      var EVSEProcessingType v_EVSEprocessing := ongoing;
      var IEC_61851_States v_state;
      var boolean v_receivedOngoing := true;
      var integer v_cnt := 0;

      if(PIXIT_SECC_DC_CableCheckOngoing == ongoing) {
         v_receivedOngoing := false;
      }

      // set state
          v_state := PICS_SECC_CMN_EvStateInChargeProgress;
          f_SECC_changeValidStateCondition(v_state);
          f_SECC_setState(v_state, v_HAL_61851_Listener);


      vc_DC_EVStatus := md_CMN_DC_EVStatus_001(true, nO_ERROR, par_EVRESSSOC);
      tc_V2G_EVCC_CableCheck_Timer.start(par_V2G_EVCC_CableCheck_Timeout);

      while(v_EVSEprocessing == ongoing and getverdict == pass){

          tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CableCheckReq);
          pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_CMN_DC_CableCheckReq_001(vc_DC_EVStatus))));

              alt {
                [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                                        md_CMN_BodyType_001(
                                                        md_CMN_DC_CableCheckRes_001(oK,finished,
                                                        md_CMN_DC_EVSEStatus_001(?,?,*,?)))))
                                                        -> value v_responseMessage {

                    v_EVSEprocessing := finished;
                    tc_V2G_EVCC_Msg_Timer.stop;
                    tc_V2G_EVCC_CableCheck_Timer.stop;
                    tc_V2G_EVCC_Ongoing_Timer.stop;
                    ongoingProcess := false;
                    v_cnt := v_cnt + 1;
                    if(ispresent(v_responseMessage.body.bodyElement.cableCheckRes.
                                 dC_EVSEStatus.eVSEIsolationStatus)) {
                        if(v_responseMessage.body.bodyElement.cableCheckRes.
                           dC_EVSEStatus.eVSEIsolationStatus != valid) {
                           logWarn("The field 'eVSEIsolationStatus' should be " &
                                     "set to 'valid'.");
                        }
                    }
                    if(v_responseMessage.body.bodyElement.cableCheckRes.
                       dC_EVSEStatus.eVSEStatusCode != eVSE_Ready) {
                       logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
                    }
                    if(v_responseMessage.body.bodyElement.cableCheckRes.
                       dC_EVSEStatus.eVSENotification != none_) {
                       logWarn("The field 'eVSENotification' should be set to 'none_'.");
                    }
                    setverdict(pass,"CableCheckRes message was correct.");
                }
                [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                                        md_CMN_BodyType_001(
                                                        md_CMN_DC_CableCheckRes_001(oK,ongoing,
                                                        md_CMN_DC_EVSEStatus_001(?,?,*,?)))))
                                                        -> value v_responseMessage {

                    tc_V2G_EVCC_Msg_Timer.stop;
                    v_receivedOngoing := true;
                    if(ispresent(v_responseMessage.body.bodyElement.cableCheckRes.
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        dC_EVSEStatus.eVSEIsolationStatus)) {
                 if(v_responseMessage.body.bodyElement.cableCheckRes.
                    dC_EVSEStatus.eVSEIsolationStatus != invalid) {
                    logWarn("The field 'eVSEIsolationStatus' should be " &
                              "set to 'invalid'.");
                 }
              }
              if(v_responseMessage.body.bodyElement.cableCheckRes.
                 dC_EVSEStatus.eVSEStatusCode != eVSE_IsolationMonitoringActive) {
                 logWarn("The field 'eVSEStatusCode' should be set to " &
                           "'eVSE_IsolationMonitoringActive'.");
              }
              if(v_responseMessage.body.bodyElement.cableCheckRes.
                 dC_EVSEStatus.eVSENotification != none_) {
                 logWarn("The field 'eVSENotification' should be set to 'none_'.");
              }
              if (ongoingProcess == false) {
                  ongoingProcess := true;
                  tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout);
              }
           }
           [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                          "misleadingly terminated " &
                                                          "by the SUT."){
              v_EVSEprocessing := finished;
              tc_V2G_EVCC_Msg_Timer.stop;
              tc_V2G_EVCC_CableCheck_Timer.stop;
              break;
           }
           [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {cableCheckRes := ?})  {
              v_EVSEprocessing := finished;
              tc_V2G_EVCC_Msg_Timer.stop;
              tc_V2G_EVCC_CableCheck_Timer.stop;
              break;
           }
           [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                        "but cannot be decoded.") {
              v_EVSEprocessing := finished;
              tc_V2G_EVCC_Msg_Timer.stop;
              tc_V2G_EVCC_CableCheck_Timer.stop;
              break;
           }
           [] a_SECC_Unexpected_Message_Content(v_vct, "Received CableCheckRes message, " &
                                                       "but with invalid data content.")  {
              v_EVSEprocessing := finished;
              tc_V2G_EVCC_Msg_Timer.stop;
              tc_V2G_EVCC_CableCheck_Timer.stop;
              break;
           }
           [] a_SECC_Timeout(v_vct){
              v_EVSEprocessing := finished;
              tc_V2G_EVCC_CableCheck_Timer.stop;
              break;
           }
           [] tc_V2G_EVCC_Ongoing_Timer.timeout
           {
              setverdict(v_vct, "Ongoing_Timer was triggered. CableCheckRes " &
                                "wasn't successful.");
              v_EVSEprocessing := finished;
              tc_V2G_EVCC_Msg_Timer.stop;
              tc_V2G_EVCC_CableCheck_Timer.stop;
              break;
           }
           []  tc_V2G_EVCC_CableCheck_Timer.timeout
           {
              setverdict(v_vct, "CableCheck_Timer was triggered. CableCheckRes " &
                                "wasn't successful.");
              v_EVSEprocessing := finished;
              tc_V2G_EVCC_Msg_Timer.stop;
              break;
           }
        }
     }
     if(PIXIT_SECC_DC_CableCheckOngoing == ongoing and not v_receivedOngoing) {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            setverdict(inconc, "The SUT did not repeat the CableCheck message sequence.");
         }
      return getverdict;
   }

function f_SECC_DC_TB_VTB_CableCheck_002(in HAL_61851_Listener v_HAL_61851_Listener)
                                        runs on SECC_Tester return verdicttype  {

      var charstring v_message;
      var boolean isShutdownOSC := false;
      var boolean isClosedTCP := false;
      var boolean isErrorMessage := false;
      var charstring v_charstring := "";
      var IEC_61851_States v_state;

      // set state
          v_state := PICS_SECC_CMN_EvStateInChargeProgress;
          f_SECC_changeValidStateCondition(v_state);
          f_SECC_setState(v_state, v_HAL_61851_Listener);
          f_SECC_setIsConfirmationFlagDC(true);
          f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                           vc_FrequencyOff);
          f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                           vc_DutyCycleOscOff);
          deactivate(vc_Default_IEC_61851_ListenerBehavior);

      tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
      tc_CMN_TCP_Connection_Termination_Timer.start(
      par_CMN_TCP_Connection_Termination_Timeout);

      pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

      alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        []    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                          fAILED_SequenceError)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                            "'fAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                       e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                      {serviceDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received CableCheckRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                            "message with responseCode 'fAILED_SequenceError' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

function f_SECC_DC_TB_VTB_CableCheck_003(in HAL_61851_Listener v_HAL_61851_Listener)
                                        runs on SECC_Tester return verdicttype  {

        var charstring v_message;
        var boolean isShutdownOSC := false;
        var boolean isClosedTCP := false;
        var boolean isErrorMessage := false;
        var charstring v_charstring := "";
        var IEC_61851_States v_state;

        // set state
            v_state := PICS_SECC_CMN_EvStateInChargeProgress;
            f_SECC_changeValidStateCondition(v_state);
            f_SECC_setState(v_state, v_HAL_61851_Listener);
            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        vc_DC_EVStatus := md_CMN_DC_EVStatus_001(true, nO_ERROR, par_EVRESSSOC);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CableCheckReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    f_rnd_SessionID(1,429496)),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_CableCheckReq_001(
                                    vc_DC_EVStatus))));

            alt {
              []
              a_Shutdown_Oscillator(pass){
                   isShutdownOSC := true;
                   if(isClosedTCP == false) {repeat;}
              }
              [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(?),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_DC_CableCheckRes_001(
                                                 fAILED_UnknownSession,?,?)))) {

                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"CableCheckRes with responseCode " &
                                "'fAILED_UnknownSession' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                              e_getPortStatus));
                isErrorMessage := true;
                repeat;
              }
              [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                            "terminated by the SUT.") {
                   isClosedTCP := true;
                   tc_CMN_TCP_Connection_Termination_Timer.stop;
                   if(isShutdownOSC == false) {
                      logWarn("SECC oscillator was not shut down before the " &
                              "TCP connection was terminated.");}
                   pt_HAL_61851_Internal_Port.clear;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                                {cableCheckRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CableCheckRes message, " &
                                                   "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a CableCheckRes " &
                            "message with responseCode 'fAILED_UnknownSession' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

  function f_SECC_DC_TB_VTB_CableCheck_004() runs on SECC_Tester return verdicttype {

      timer t_wait;
      var boolean isShutdownOSC := false;
      var boolean isClosedTCP := false;
      var float v_timeShift := 1.0;

      // Wait until sequence timeout in SUT should expire
      log("Proccessing time: " & toString<float>(vc_processingTime) & "s.");
      t_wait.start(par_V2G_SECC_Sequence_Timeout -
                   v_timeShift - vc_processingTime);
      pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                  e_getPortStatus));

      alt {
          [] t_wait.timeout;
          [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                          "terminated by the SUT " &
                                          "before timeout has expired.");
          [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
              setverdict(fail, "No V2G message was expected.");
          }
      }

      if(getverdict == pass) {

              f_SECC_setIsConfirmationFlagDC(true);
              f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                               vc_FrequencyOff);
              f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                               vc_DutyCycleOscOff);
              deactivate(vc_Default_IEC_61851_ListenerBehavior);

          sleep(v_timeShift - par_CMN_Transmission_Delay);

          tc_CMN_TCP_Connection_Termination_Timer.start(
          par_CMN_TCP_Connection_Termination_Timeout);

          alt {
              []
              a_Shutdown_Oscillator(pass){
                    isShutdownOSC := true;
                    if(isClosedTCP == false) {repeat;}
              }
              [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
                   isClosedTCP := true;
                   tc_CMN_TCP_Connection_Termination_Timer.stop;
                   if(isShutdownOSC == false) {
                      logWarn("SECC oscillator was not shut down before the " &
                              "TCP connection was terminated.");}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection was " &
                                                       ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_CableCheck_005() runs on SECC_Tester return verdicttype {

      var charstring v_message;
      var boolean isShutdownOSC := false;
      var boolean isClosedTCP := false;
      var boolean isErrorMessage := false;
      var charstring v_charstring := "";

      // set state
          f_SECC_setIsConfirmationFlagDC(true);
          f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                           vc_FrequencyOff);
          f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                           vc_DutyCycleOscOff);
          deactivate(vc_Default_IEC_61851_ListenerBehavior);

      vc_DC_EVStatus := md_CMN_DC_EVStatus_001(true, nO_ERROR, par_EVRESSSOC);

      tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CableCheckReq);
      tc_CMN_TCP_Connection_Termination_Timer.start(
      par_CMN_TCP_Connection_Termination_Timeout);

      pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(
                                  vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_CableCheckReq_001(
                                  vc_DC_EVStatus))));

          alt {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(
                                            vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_CMN_DC_CableCheckRes_001(
                                            fAILED,?,?)))) {

                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"CableCheckRes with responseCode " &
                                "'fAILED' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                          e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        {cableCheckRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CableCheckRes message, " &
                                                       "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a CableCheckRes " &
                            "message with responseCode 'fAILED' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }
}
```

I.2.2.15 SECC functions for PreCharge

```
module TestBehavior_SECC_PreCharge {

    import from Timer_15118_2 all;
    import from ComponentsAndPorts all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from Templates_CMN_PreCharge all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_CableCheck all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from Configurations_15118_2 all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Pics_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from DataStructure_HAL_61851 all;
    import from LibFunctions_15118_2 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Pixit_15118 all;
    import from Timer_15118 all;

    function f_SECC_DC_TB_VTB_PreCharge_001(in verdicttype v_vct)
                                        runs on SECC_Tester
                                        return verdicttype {

        var V2G_Message v_responseMessage;
        var charstring v_message;
        var integer v_cnt := 0;

        tc_V2G_EVCC_PreCharge_Timer.start(par_V2G_EVCC_PreCharge_Timeout);

        vc_EVSEPresentVoltage := { multiplier := 0, unit := v, value_ := 0 };
        log("vc_EVTargetVoltage: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                                            vc_EVTargetVoltage)));

        var float v_deviation :=
        (f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage) / 100.0) * par_ValidDeviationPerCent;
        var float v_EVTargetVoltageValidMaxDeviation :=
        f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage) + v_deviation;
        var float v_EVTargetVoltageValidMinDeviation :=
        f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage) - v_deviation;

        do {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PreChargeReq);
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                              md_CMN_MessageHeaderType_001(vc_SessionID),
                              md_CMN_BodyType_001(
                              md_CMN_DC_PreChargeReq_001(
                              vc_DC_EVStatus, vc_EVTargetVoltage,
                              par_CurrentPreCharge))));

  alt {
      [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_PreChargeRes_001(oK,
                                        md_CMN_DC_EVSEStatus_001(?,?,*,?),?))))
                                        -> value v_responseMessage {
          tc_V2G_EVCC_Msg_Timer.stop;

          vc_EVSEPresentVoltage := v_responseMessage.body.bodyElement.
                                   preChargeRes.eVSEPresentVoltage;
          f_check_PhysicalValue_valueRange(cc_eVSEPresentVoltage,
                                           vc_EVSEPresentVoltage, v,
                                           cc_maximum_value_physicalValue_case2,
                                         v_vct);
          log("vc_EVSEPresentVoltage: " & toString<float>(
          f_CMN_PhysicalValue_GetValue(vc_EVSEPresentVoltage)));

          if(ispresent(v_responseMessage.body.bodyElement.preChargeRes.
                       dC_EVSEStatus.eVSEIsolationStatus)) {
              if(v_responseMessage.body.bodyElement.preChargeRes.dC_EVSEStatus.
                 eVSEIsolationStatus != valid) {
                 logWarn("The field 'eVSEIsolationStatus' should be set to 'valid'.");
              }
          }
          if(v_responseMessage.body.bodyElement.preChargeRes.dC_EVSEStatus.
             eVSEStatusCode != eVSE_Ready) {
             logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
          }
          if(v_responseMessage.body.bodyElement.preChargeRes.dC_EVSEStatus.
             eVSENotification != none_) {
             logWarn("The field 'eVSENotification' should be set to 'none_'.");
          }
          if(getverdict == pass) {
               setverdict(pass,"PreChargeRes message was correct.");
          }
      }
      [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                     "was misleadingly " &
                                                     "terminated by the SUT."){
         tc_V2G_EVCC_Msg_Timer.stop;
         tc_V2G_EVCC_PreCharge_Timer.stop;
         v_EVTargetVoltageValidMinDeviation := 0.0;
      }
      [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                      {preChargeRes := ?}) {
         tc_V2G_EVCC_Msg_Timer.stop;
         tc_V2G_EVCC_PreCharge_Timer.stop;
         v_EVTargetVoltageValidMinDeviation := 0.0;
      }
      [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                   "but cannot be decoded.") {
         tc_V2G_EVCC_Msg_Timer.stop;
         tc_V2G_EVCC_PreCharge_Timer.stop;
         v_EVTargetVoltageValidMinDeviation := 0.0;
      }
      [] a_SECC_Unexpected_Message_Content(v_vct, "Received PreChargeRes message, " &
                                                  "but with invalid data content.") {
         tc_V2G_EVCC_Msg_Timer.stop;
         tc_V2G_EVCC_PreCharge_Timer.stop;
         v_EVTargetVoltageValidMinDeviation := 0.0;
      }
      [] a_SECC_Timeout(v_vct) {
         tc_V2G_EVCC_PreCharge_Timer.stop;
         v_EVTargetVoltageValidMinDeviation := 0.0;
      }
      [] tc_V2G_EVCC_PreCharge_Timer.timeout {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                tc_V2G_EVCC_Msg_Timer.stop;
                v_EVTargetVoltageValidMinDeviation := 0.0;
                setverdict(v_vct, "PreChargeTimer was triggered. " &
                                  "PreCharge wasn't successful.");
            }
        }
    } while((f_CMN_PhysicalValue_GetValue(vc_EVSEPresentVoltage) <
            v_EVTargetVoltageValidMinDeviation or
            f_CMN_PhysicalValue_GetValue(vc_EVSEPresentVoltage) >
            v_EVTargetVoltageValidMaxDeviation) and
            (getverdict == pass));

    vc_EVTargetVoltage := fx_setPhysicalValueType(
    f_CMN_PhysicalValue_GetValue(vc_EVMaximumVoltageLimit) - 10.0, "v");
    log("AFTER vc_EVTargetVoltage: " & toString<float>(
    f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage)));

    return getverdict;
 }

function f_SECC_DC_TB_VTB_PreCharge_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                          fAILED_SequenceError)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                            "'fAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {serviceDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                 "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received PreChargeRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                        "message with responseCode 'fAILED_SequenceError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_PreCharge_003() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_PreChargeReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(
                                f_rnd_SessionID(1,429496)),
                                md_CMN_BodyType_001(
                                md_CMN_DC_PreChargeReq_001(
                                vc_DC_EVStatus,vc_EVTargetVoltage,
                                vc_EVTargetCurrent))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(?),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_PreChargeRes_001(
                                          fAILED_UnknownSession,?,?)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"PreChargeRes with responseCode " &
                            "'fAILED_UnknownSession' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                         {preChargeRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received PreChargeRes message, " &
                                               "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a PreChargeRes " &
                        "message with responseCode 'fAILED_UnknownSession' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_PreCharge_004() runs on SECC_Tester return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var float v_timeShift := 1.0;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
                 v_timeShift);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                        "terminated by the SUT " &
                                        "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                    "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                          "but TCP connection was " &
                                                          ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout() {
                    setverdict(fail,"SUT has not terminated the TCP connection " &
                                    "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }
}
```

I.2.2.16 SECC functions for CurrentDemand

```
module TestBehavior_SECC_CurrentDemand {

    import from Timer_15118_2 all;
    import from ComponentsAndPorts {type SECC_Tester;};
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" {type V2G_Message;}
    import from TestBehavior_SECC_CommonBehavior all;
    import from Templates_CMN_CurrentDemand all;
    import from Templates_SECC_ServiceDiscovery all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage all;
    import from DataStructure_HAL_61851 all;
    import from TTlibrary_Logging all;
    import from Pics_15118 all;
    import from Pics_15118_2 all;
    import from Pixit_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from LibFunctions_15118_2 all;
    import from Pixit_15118 all;
    import from Timer_15118 all;

    function f_SECC_DC_TB_VTB_CurrentDemand_001(in verdicttype v_vct)
                                                runs on SECC_Tester
                                                return verdicttype {

        var V2G_Message v_responseMessage;
        var charstring v_message;

            log("------------------------------------------------------------");
            log("EV maximum charging limits:");
            log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumCurrentLimit)));
            log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumVoltageLimit)));
            log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumPowerLimit)));
            log("------------------------------------------------------------");

              tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CurrentDemandReq);
            pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(md_CMN_DC_CurrentDemandReq_001(
                                        vc_DC_EVStatus,vc_EVTargetCurrent,
                                        vc_EVMaximumVoltageLimit,
                                        vc_EVMaximumCurrentLimit,vc_EVMaximumPowerLimit,
                                        false, true ,
                                        vc_remainingTimeToFullSoC,
                                        vc_remainingTimeToBulkSoC,
                                        vc_EVTargetVoltage))));

              alt {
                  [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(vc_SessionID),
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_CurrentDemandRes_001(
                                    oK,md_CMN_DC_EVSEStatus_001(?,?,*,?),
                                    ?,?,?,?,?,*,*,*,?,?,*,*))))
                                    -> value v_responseMessage {
  tc_V2G_EVCC_Msg_Timer.stop;
vc_EVSEPresentVoltage  := v_responseMessage.body.bodyElement.
                          currentDemandRes.eVSEPresentVoltage;
f_check_PhysicalValue_valueRange(cc_eVSEPresentVoltage,
                                 vc_EVSEPresentVoltage, v,
                                 cc_maximum_value_physicalValue_case2,
                                 v_vct);
vc_EVSEPresentCurrent  := v_responseMessage.body.bodyElement.
                          currentDemandRes.eVSEPresentCurrent;
f_check_PhysicalValue_valueRange(cc_eVSEPresentCurrent,
                                 vc_EVSEPresentCurrent, a,
                                 cc_maximum_value_physicalValue_case1,
                                 v_vct);
if(ispresent(v_responseMessage.body.bodyElement.
             currentDemandRes.eVSEMaximumCurrentLimit)) {
    vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
                                  currentDemandRes.eVSEMaximumCurrentLimit;
    f_check_PhysicalValue_valueRange(cc_eVSEMaximumCurrentLimit,
                                     vc_EVSEMaximumCurrentLimit, a,
                                     cc_maximum_value_physicalValue_case1,
                                     v_vct);
}
if(ispresent(v_responseMessage.body.bodyElement.
             currentDemandRes.eVSEMaximumVoltageLimit)) {
    vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
                                  currentDemandRes.eVSEMaximumVoltageLimit;
    f_check_PhysicalValue_valueRange(cc_eVSEMaximumVoltageLimit,
                                     vc_EVSEMaximumVoltageLimit, v,
                                     cc_maximum_value_physicalValue_case2,
                                     v_vct);
}
if(ispresent(v_responseMessage.body.bodyElement.
             currentDemandRes.eVSEMaximumPowerLimit)) {
    vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
                                currentDemandRes.eVSEMaximumPowerLimit;
    f_check_PhysicalValue_valueRange(cc_eVSEMaximumPowerLimit,
                                     vc_EVSEMaximumPowerLimit, w,
                                     cc_maximum_value_physicalValue_case4,
                                     v_vct);
}
vc_EVSENotification := v_responseMessage.body.bodyElement.
                       currentDemandRes.dC_EVSEStatus.eVSENotification;
vc_EVSEStatusCode := v_responseMessage.body.bodyElement.
                       currentDemandRes.dC_EVSEStatus.eVSEStatusCode;

if(vc_EVSEStatusCode == eVSE_Shutdown and
   not vc_EVSENotification == stopCharging) {
    logWarn("The field EVSEStatusCode is set to 'eVSE_Shutdown', " &
            "but the field EVSENotification is not set to " &
            "'stopCharging'. An EVSE-initiated " &
            "shutdown shall be triggered with " &
            "'stopCharging' according to [V2G2-679].");
}
vc_EvseId := v_responseMessage.body.bodyElement.currentDemandRes.eVSEID;
if(ispresent(v_responseMessage.body.bodyElement.currentDemandRes.
             meterInfo)) {
    vc_MeterInfo := v_responseMessage.body.bodyElement.
                    currentDemandRes.meterInfo;
}
if(ispresent(v_responseMessage.body.bodyElement.currentDemandRes.
             dC_EVSEStatus.eVSEIsolationStatus)) {
    if(v_responseMessage.body.bodyElement.currentDemandRes.
       dC_EVSEStatus.eVSEIsolationStatus != valid) {
       logWarn("The field 'eVSEIsolationStatus' should be set to 'valid'.");
    }
}
if(v_responseMessage.body.bodyElement.currentDemandRes.dC_EVSEStatus.
   eVSEStatusCode != eVSE_Ready) {
   logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    log("-------------------------------------------------------------");
                    log("EVSE maximum charging limits:");
                    log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                        vc_EVSEMaximumCurrentLimit)));
                    log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                        vc_EVSEMaximumVoltageLimit)));
                    log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                        vc_EVSEMaximumPowerLimit)));
                    log("-------------------------------------------------------------");
                    if(vc_EvseId != cc_EvseId_NotSup) {
                        var boolean result := fx_validateEVSEID(vc_EvseId, "ISO");
                        if(not result) {
                                setverdict(v_vct,"Invalid EVSEID syntax was detected.");
                        }
                    }
                    if(ispresent(v_responseMessage.body.bodyElement.
                                 currentDemandRes.receiptRequired)) {
                        vc_receiptRequired  := v_responseMessage.body.bodyElement.
                                               currentDemandRes.receiptRequired;
                        if(PICS_CMN_CMN_IdentificationMode == eIM) {
                            if(not(vc_receiptRequired == false)){
                                    setverdict(v_vct, "ReceiptRequired  shall always set " &
                                                      "to false in EIM Identification Mode.");
                            }
                        }
                        else {
                            if(PIXIT_SECC_CMN_ReceiptRequired == receiptRequired) {
                                if(vc_receiptRequired == false){
                                        setverdict(v_vct, "ReceiptRequired  shall always set " &
                                                          "to true if the SECC requires a " &
                                                          "metering receipt from the EVCC.");
                                }
                            }
                            else if(PIXIT_SECC_CMN_ReceiptRequired == none_){
                                if(vc_receiptRequired == true){
                                        setverdict(v_vct, "ReceiptRequired  shall always set " &
                                                          "to false if the SECC not requires a " &
                                                          "metering receipt from the EVCC.");
                                }
                            }
                        }
                    }
                    if(getverdict == pass) {
                        setverdict(pass,"CurrentDemandRes message was correct.");
                    }
                }
                [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                               "was misleadingly " &
                                                               "terminated by the SUT.");
                [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                                 {currentDemandRes := ?});
                [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                             "but cannot be decoded.");
                [] a_SECC_Unexpected_Message_Content(v_vct, "Received CurrentDemandRes message, " &
                                                            "but with invalid data content.");
                [] a_SECC_Timeout(v_vct);
            }
        return getverdict;
    }

function f_SECC_DC_TB_VTB_CurrentDemand_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

        alt {
            []
            a_Shutdown_Oscillator(pass){
                 isShutdownOSC := true;
                 if(isClosedTCP == false) {repeat;}
            }
            []    pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
                                              md_CMN_BodyType_001(
                                              mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                              fAILED_SequenceError)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                               "'fAILED_SequenceError' is successful.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                         e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {serviceDiscoveryRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received CurrentDemandRes message, " &
                                                       "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                            "message with responseCode 'fAILED_SequenceError' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    function f_SECC_DC_TB_VTB_CurrentDemand_003() runs on SECC_Tester return verdicttype {

        var charstring v_message;
        var boolean isShutdownOSC := false;
        var boolean isClosedTCP := false;
        var boolean isErrorMessage := false;
        var charstring v_charstring := "";

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CurrentDemandReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(
                                f_rnd_SessionID(1,429496)),
                                md_CMN_BodyType_001(
                                md_CMN_DC_CurrentDemandReq_001(
                                vc_DC_EVStatus,vc_EVTargetCurrent,
                                vc_EVMaximumVoltageLimit,vc_EVMaximumCurrentLimit,
                                vc_EVMaximumPowerLimit, false, true,
                                vc_remainingTimeToFullSoC,
                                vc_remainingTimeToBulkSoC,
                                vc_EVTargetVoltage))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
             isShutdownOSC := true;
             if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(?),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_CurrentDemandRes_001(
                                          fAILED_UnknownSession,?,?,?,?,?,?,
                                          omit,omit,omit,?,?,omit,omit)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"CurrentDemandRes with responseCode " &
                            "'fAILED_UnknownSession' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {currentDemandRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received CurrentDemandRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a CurrentDemandRes " &
                        "message with responseCode 'fAILED_UnknownSession' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_CurrentDemand_004() runs on SECC_Tester return verdicttype {

    timer t_wait;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var float v_timeShift := 1.0;

    // Wait until sequence timeout in SUT should expire
    t_wait.start(par_V2G_SECC_Sequence_Timeout -
                 v_timeShift);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                        "terminated by the SUT " &
                                        "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                   "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                 "but TCP connection was " &
                                                 ">>NOT<< terminated by the SUT");
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_CurrentDemand_005() runs on SECC_Tester return verdicttype  {

    var V2G_Message v_responseMessage;
    var charstring v_message;

        log("-------------------------------------------------------------");
        log("EV maximum charging limits:");
        log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumCurrentLimit)));
        log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumVoltageLimit)));
        log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumPowerLimit)));
        log("-------------------------------------------------------------");

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CurrentDemandReq);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                              md_CMN_MessageHeaderType_001(vc_SessionID),
                              md_CMN_BodyType_001(md_CMN_DC_CurrentDemandReq_001(
                              vc_DC_EVStatus,vc_EVTargetCurrent,
                              vc_EVMaximumVoltageLimit,
                              vc_EVMaximumCurrentLimit,vc_EVMaximumPowerLimit,
                              false, true ,
                              vc_remainingTimeToFullSoC,
                              vc_remainingTimeToBulkSoC,
                              vc_EVTargetVoltage))));

alt {
    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_DC_CurrentDemandRes_001(
                                      oK,md_CMN_DC_EVSEStatus_001(?,?,*,?),
                                      ?,?,?,?,?,*,*,*,?,?,?,*))))
                                      -> value v_responseMessage {
        tc_V2G_EVCC_Msg_Timer.stop;
        vc_EVSEPresentVoltage  := v_responseMessage.body.bodyElement.
                                  currentDemandRes.eVSEPresentVoltage;
        f_check_PhysicalValue_valueRange(cc_eVSEPresentVoltage,
                                         vc_EVSEPresentVoltage, v,
                                         cc_maximum_value_physicalValue_case2,
                                         fail);
        vc_EVSEPresentCurrent  := v_responseMessage.body.bodyElement.
                                  currentDemandRes.eVSEPresentCurrent;
        f_check_PhysicalValue_valueRange(cc_eVSEPresentCurrent,
                                         vc_EVSEPresentCurrent, a,
                                         cc_maximum_value_physicalValue_case1,
                                         fail);
        vc_EVSENotification := v_responseMessage.body.bodyElement.
                               currentDemandRes.dC_EVSEStatus.eVSENotification;
        vc_EvseId := v_responseMessage.body.bodyElement.currentDemandRes.eVSEID;
        vc_MeterInfo := v_responseMessage.body.bodyElement.currentDemandRes.
                        meterInfo;
        if(ispresent(v_responseMessage.body.bodyElement.currentDemandRes.
                     dC_EVSEStatus.eVSEIsolationStatus)) {
            if(v_responseMessage.body.bodyElement.currentDemandRes.
               dC_EVSEStatus.eVSEIsolationStatus != valid) {
               logWarn("The field 'eVSEIsolationStatus' should be " &
                         "set to 'valid'.");
            }
        }
        if(v_responseMessage.body.bodyElement.currentDemandRes.dC_EVSEStatus.
           eVSEStatusCode != eVSE_Ready) {
           logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
        }
        if(vc_EvseId != cc_EvseId_NotSup) {
            var boolean result := fx_validateEVSEID(vc_EvseId, "ISO");
            if(not result) {
                setverdict(fail,"Invalid EVSEID syntax was detected.");
            }
        }
        if(ispresent(v_responseMessage.body.bodyElement.
                     currentDemandRes.receiptRequired)) {
            vc_receiptRequired  := v_responseMessage.body.bodyElement.
                                   currentDemandRes.receiptRequired;
            if(PICS_CMN_CMN_IdentificationMode == eIM) {
                if(not(vc_receiptRequired == false)){
                    setverdict(fail, "ReceiptRequired  shall always set " &
                                     "to false in EIM Identification Mode.");
                }
            }
            else {
                if(PIXIT_SECC_CMN_ReceiptRequired == receiptRequired) {
                    if(vc_receiptRequired == false){
                        setverdict(fail, "ReceiptRequired  shall always set " &
                                         "to true if the SECC requires a " &
                                         "metering receipt from the EVCC");
                    }
                }
                else if(PIXIT_SECC_CMN_ReceiptRequired == none_){
                    if(vc_receiptRequired == true){
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                                    setverdict(fail, "ReceiptRequired  shall always set " &
                                                     "to false if the SECC not requires a " &
                                                     "metering receipt from the EVCC");
                                }
                            }
                        }
                    }
                    if(ispresent(vc_MeterInfo.meterReading)) {
                         if(vc_MeterInfo.meterReading < 0 or
                            vc_MeterInfo.meterReading > cc_maximum_value_meterReading) {
                             setverdict(fail, "Invalid value range for MeterReading detected.");
                         }
                    }
                    if(getverdict == pass) {
                        setverdict(pass,"CurrentDemandRes message was correct.");
                    }
                }
                [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                              "was misleadingly " &
                                                              "terminated by the SUT.");
                [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                                 {currentDemandRes := ?});
                [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                            "but cannot be decoded.");
                [] a_SECC_Unexpected_Message_Content(fail, "Received CurrentDemandRes message, " &
                                                           "but with invalid data content.");
                [] a_SECC_Timeout(fail);
            }

        return getverdict;
    }

    function f_checkPresentChargeParameter(in PhysicalValueType vc_EVSEPresentCurrent,
                                           in PhysicalValueType vc_EVSEPresentVoltage,
                                           in PhysicalValueType vc_EVMaximumCurrentLimit,
                                           in PhysicalValueType vc_EVMaximumVoltageLimit,
                                           in verdicttype v_vct) {

         if (f_CMN_PhysicalValue_GetValue(vc_EVSEPresentCurrent) >
              f_CMN_PhysicalValue_GetValue(vc_EVMaximumCurrentLimit)) {
            setverdict(v_vct, "The present current is greater than the EV maximum current limit. " &
                              "The charging process will be terminated.");
         }
         if    (f_CMN_PhysicalValue_GetValue(vc_EVSEPresentVoltage) >
              f_CMN_PhysicalValue_GetValue(vc_EVMaximumVoltageLimit)) {
             setverdict(v_vct, "The present voltage is greater than the EV maximum voltage limit. " &
                               "The charging process will be terminated.");
         }
    }
}
```

I.2.2.17 SECC functions for WeldingDetection

```
module TestBehavior_SECC_WeldingDetection {

      import from Timer_15118_2 all;
      import from ComponentsAndPorts all;
      import from DataStructure_HAL_61851 all;
      import from TestBehavior_SECC_CommonBehavior all;
      import from Templates_CMN_V2GMessage all;
      import from urn_iso_15118_2_2013_MsgDef all;
      import from LibFunctions_15118_2 all;
      import from Templates_CMN_ServiceDiscovery all;
      import from Templates_SECC_ServiceDiscovery all;
      import from Templates_CMN_WeldingDetection all;
      import from Templates_CMN_CableCheck all;
      import from TTlibrary_Logging all;
      import from Pics_15118 all;
      import from Pics_15118_2 all;
      import from Services_HAL_61851_1 all;
      import from Templates_CMN_V2GTcpTlsPortControl all;
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Timer_15118 all;

import from Pixit_15118 all;

function f_SECC_DC_TB_VTB_WeldingDetection_001(in verdicttype v_vct)
                                                runs on SECC_Tester
                                                return verdicttype {

    var V2G_Message v_responseMessage;
    var charstring v_message;
    timer t_EVCC_WeldingDetection;
    var boolean v_reducedSECCVoltage := false;
    var float v_chargingEndVoltage := f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage) - 10.0;
    var float v_deviation := (v_chargingEndVoltage / 100.0) * par_ValidDeviationPerCent;
    var float v_voltageValidMaxDeviation := v_chargingEndVoltage + v_deviation;
    var float v_voltageValidMinDeviation := v_chargingEndVoltage - v_deviation;
    var boolean v_weldingDetectionFinished := false;
    var float v_currentStart := 1.0;

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_WeldingDetectionReq);
    t_EVCC_WeldingDetection.start(par_EVCC_WeldingDetection_Timeout);
    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_WeldingDetectionReq_001(vc_DC_EVStatus))));

    alt {
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_WeldingDetectionRes_001(oK,
                                        md_CMN_DC_EVSEStatus_001(?,?,*,?),?))))
                                        -> value v_responseMessage {

            vc_EVSEPresentVoltage := v_responseMessage.body.bodyElement.
                                     weldingDetectionRes.eVSEPresentVoltage;
            f_check_PhysicalValue_valueRange(cc_eVSEPresentVoltage,
                                             vc_EVSEPresentVoltage, v,
                                             cc_maximum_value_physicalValue_case2,
                                           v_vct);
            tc_V2G_EVCC_Msg_Timer.stop;
            if(ispresent(v_responseMessage.body.bodyElement.
                         weldingDetectionRes.dC_EVSEStatus.eVSEIsolationStatus)) {
                if(v_responseMessage.body.bodyElement.weldingDetectionRes.dC_EVSEStatus.
                   eVSEIsolationStatus != valid) {
                   logWarn("The field 'eVSEIsolationStatus' should be set to 'valid'.");
                }
            }
            if(v_responseMessage.body.bodyElement.weldingDetectionRes.dC_EVSEStatus.
               eVSEStatusCode != eVSE_Ready) {
               logWarn("The field 'eVSEStatusCode' should be set to 'eVSE_Ready'.");
            }
            if(v_responseMessage.body.bodyElement.weldingDetectionRes.dC_EVSEStatus.
               eVSENotification != none_) {
               logWarn("The field 'eVSENotification' should be set to 'none_'.");
            }
            if(getverdict == pass) {
              setverdict(pass," WeldingDetectionRes message was correct.");
            }
        }
        [] a_SECC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                       "was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {weldingDetectionRes := ?});
        [isError()] a_SECC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(v_vct, "Received WeldingDetectionRes message, " &
                                                      "but with invalid data content.");
        [] a_SECC_Timeout(v_vct);
        [] t_EVCC_WeldingDetection.timeout {
            setverdict(inconc, "The Welding Detection process could not be executed " &
                               "within 'par_EVCC_WeldingDetection_Timeout'.");
        }
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        return getverdict;
    }

function f_SECC_DC_TB_VTB_WeldingDetection_002() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryReq_002(omit))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          mdw_SECC_CMN_ServiceDiscoveryRes_001(
                                          fAILED_SequenceError)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"ServiceDiscoveryRes with responseCode " &
                            "'fAILED_SequenceError' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                       {serviceDiscoveryRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received WeldingDetectionRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a ServiceDiscoveryRes " &
                        "message with responseCode 'fAILED_SequenceError' " &
                        "or other message data content was not conform. ";
    }
    if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
       setverdict(fail,v_charstring);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }
    return getverdict;
}

function f_SECC_DC_TB_VTB_WeldingDetection_003() runs on SECC_Tester return verdicttype {

    var charstring v_message;
    var boolean isShutdownOSC := false;
    var boolean isClosedTCP := false;
    var boolean isErrorMessage := false;
    var charstring v_charstring := "";

        f_SECC_setIsConfirmationFlagDC(true);
        f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                         vc_FrequencyOff);
        f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                         vc_DutyCycleOscOff);
        deactivate(vc_Default_IEC_61851_ListenerBehavior);

    tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_WeldingDetectionReq);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(
                                 f_rnd_SessionID(1,429496)),
                                 md_CMN_BodyType_001(
                                 md_CMN_DC_WeldingDetectionReq_001(
                                 vc_DC_EVStatus))));

    alt {
        []
        a_Shutdown_Oscillator(pass){
              isShutdownOSC := true;
              if(isClosedTCP == false) {repeat;}
        }
        [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(?),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_WeldingDetectionRes_001(
                                          fAILED_UnknownSession,?,?)))) {
            tc_V2G_EVCC_Msg_Timer.stop;
            setverdict(pass,"WeldingDetectionRes with responseCode " &
                            "'fAILED_UnknownSession' is successful.");

            pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                        e_getPortStatus));
            isErrorMessage := true;
            repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
             isClosedTCP := true;
             tc_CMN_TCP_Connection_Termination_Timer.stop;
             if(isShutdownOSC == false) {
                logWarn("SECC oscillator was not shut down before the " &
                        "TCP connection was terminated.");}
             pt_HAL_61851_Internal_Port.clear;
        }
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                       {weldingDetectionRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received WeldingDetectionRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
        [] a_SECC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
        }
    }
    if(not isErrorMessage) {
        v_charstring := v_charstring & "SUT has not sent a WeldingDetectionRes " &
                        "message with responseCode 'fAILED_UnknownSession' " &
                        "or other message data content was not conform. ";
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    function f_SECC_DC_TB_VTB_WeldingDetection_004() runs on SECC_Tester return verdicttype {

        timer t_wait;
        var boolean isShutdownOSC := false;
        var boolean isClosedTCP := false;
        var float v_timeShift := 1.0;

        // Wait until sequence timeout in SUT should expire
        t_wait.start(par_V2G_SECC_Sequence_Timeout -
                     v_timeShift);
        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));

        alt {
            [] t_wait.timeout;
            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                          "terminated by the SUT " &
                                          "before timeout has expired.");
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive {
                setverdict(fail, "No V2G message was expected.");
            }
        }

        if(getverdict == pass) {

                f_SECC_setIsConfirmationFlagDC(true);
                f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                                 vc_FrequencyOff);
                f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                                 vc_DutyCycleOscOff);
                deactivate(vc_Default_IEC_61851_ListenerBehavior);

            sleep(v_timeShift - par_CMN_Transmission_Delay);

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                []
                a_Shutdown_Oscillator(pass){
                      isShutdownOSC := true;
                      if(isClosedTCP == false) {repeat;}
                }
                [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                        "terminated by the SUT.") {
                     isClosedTCP := true;
                     tc_CMN_TCP_Connection_Termination_Timer.stop;
                     if(isShutdownOSC == false) {
                        logWarn("SECC oscillator was not shut down before the " &
                                "TCP connection was terminated.");}
                     pt_HAL_61851_Internal_Port.clear;
                }
                [] a_SECC_Unexpected_Message_Content(fail, "Received any message, " &
                                                         "but TCP connection was " &
                                                         ">>NOT<< terminated by the SUT");
                [] a_SECC_TCP_Connection_Termination_Timeout() {
                    setverdict(fail,"SUT has not terminated the TCP connection " &
                                    "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_SECC_DC_TB_VTB_WeldingDetection_005() runs on SECC_Tester return verdicttype {

        var charstring v_message;
        var boolean isShutdownOSC := false;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        var boolean isClosedTCP := false;
        var boolean isErrorMessage := false;
        var charstring v_charstring := "";

            f_SECC_setIsConfirmationFlagDC(true);
            f_SECC_changeValidFrequencyRange(vc_FrequencyOff,
                                             vc_FrequencyOff);
            f_SECC_changeValidDutyCycleRange(vc_DutyCycleOscOff,
                                             vc_DutyCycleOscOff);
            deactivate(vc_Default_IEC_61851_ListenerBehavior);

        tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_WeldingDetectionReq);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_WeldingDetectionReq_001(
                                  vc_DC_EVStatus))));

        alt {
            []
            a_Shutdown_Oscillator(pass){
                  isShutdownOSC := true;
                  if(isClosedTCP == false) {repeat;}
            }
            [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_DC_WeldingDetectionRes_001(
                                                fAILED,?,?)))) {
                tc_V2G_EVCC_Msg_Timer.stop;
                setverdict(pass,"WeldingDetectionRes with responseCode 'fAILED' is successful. " &
                                "SUT could not measuring state B before.");

                pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                             e_getPortStatus));
                isErrorMessage := true;
                repeat;
            }
            [] a_SECC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                 isClosedTCP := true;
                 tc_CMN_TCP_Connection_Termination_Timer.stop;
                 if(isShutdownOSC == false) {
                    logWarn("SECC oscillator was not shut down before the " &
                            "TCP connection was terminated.");}
                 pt_HAL_61851_Internal_Port.clear;
            }
            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {weldingDetectionRes := ?});
            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_SECC_Unexpected_Message_Content(fail, "Received WeldingDetectionRes message, " &
                                                       "but with invalid data content.");
            [] a_SECC_Timeout(fail);
            [] a_SECC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            }
        }
        if(not isErrorMessage) {
            v_charstring := v_charstring & "SUT has not sent a WeldingDetectionRes " &
                            "message with responseCode 'fAILED' " &
                            "or other message data content was not conform. ";
        }
        if(not(isClosedTCP and isErrorMessage) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }
}
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

I.3 EVCC functions

I.3.1 V2G protocol handshake

I.3.1.1 EVCC functions for SupportedAppProtocol

```
module TestBehavior_EVCC_SupportedAppProtocol {

    import from Pics_15118_2 all;
    import from Pics_15118 all;
    import from Timer_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from Templates_CMN_SupportedAppProtocol all;
    import from Templates_EVCC_SupportedAppProtocol all;
    import from Templates_CMN_SessionSetup all;
    import from Templates_CMN_SDP all;
    import from DataStructure_SDP all;
    import from Templates_CMN_V2GTPMessage all;
    import from ComponentsAndPorts all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from urn_iso_15118_2_2010_AppProtocol all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Templates_CMN_SessionSetup all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from DataStructure_V2G_TCP_TLS_Port_Control all;
    import from Timer_15118 all;

    function f_EVCC_CMN_TB_VTB_SupportedAppProtocol_001(in verdicttype v_vct)
                                                        runs on EVCC_Tester
                                                        return verdicttype {

        var SupportedAppProtocolReq v_requestMessage;
        var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_responseMessage;
        var V2G_TCP_TLS_Port_Control_Ack_TYPE ackType;
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

         pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Response_TYPE_001(
                                  par_SECC_IPaddress, vc_V2G_Port_PortNumber,
                                  vc_Security, vc_TransportProtocol));

        alt {
            //Handling SDP if more than one from SUT is received
            []pt_V2G_UDP_SDP_Port.receive(md_CMN_SDP_Request_TYPE_001(?, ?)) {

                 pt_V2G_UDP_SDP_Port.send(md_CMN_SDP_Response_TYPE_001(
                                          par_SECC_IPaddress, vc_V2G_Port_PortNumber,
                                          vc_Security, vc_TransportProtocol));
                 repeat;
            }
            [] pt_V2G_TCP_TLS_Port_Control.receive(md_CMN_CMN_tcpTlsStatusRes_001(
                                                e_connectPort,?,?,*))
                                                -> value v_responseMessage {

                var boolean isClosed := v_responseMessage.result.portStatus.closed;
                ackType := v_responseMessage.ack;
                if(ackType == ACK_SUCCESS) {
                   setverdict(pass, "TCP or TLS handshake was successful.");
                   repeat;
                } else if(ackType == ACK_FAILED) {
                   setverdict(v_vct, "TCP or TLS handshake was not successful.");
                }
            }
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(mw_EVCC_CMN_SupportedAppProtocolReq_001)
                                              -> value v_requestMessage {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            tc_V2G_SECC_Sequence_Timer.stop;
            f_check_Supported_ISO_Namespace(v_requestMessage, v_vct);
            if(getverdict != pass) {
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_CMN_SupportedAppProtocolRes_002(
                                                  failed_NoNegotiation));
            }
            else {
                setverdict(pass,"SupportedAppProtocolReq message was correct.");
            }
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                        "terminated by the SUT.");
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received SupportedAppProtocolReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_SupportedAppProtocol_002() runs on EVCC_Tester
                                                       return verdicttype {

    var V2G_Message v_requestMessage;

      f_EVCC_setPwmMode(e_PosVolt12);


    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_CMN_SupportedAppProtocolRes_002(
                                 failed_NoNegotiation));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(?))) {

            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"V2G message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_SupportedAppProtocol_003() runs on EVCC_Tester
                                                       return verdicttype {

    timer t_wait;

    // Wait until SUT message timeout
    t_wait.start(par_V2G_EVCC_Msg_Timeout_SupportedAppProtocolReq -
                 par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }

        if(getverdict == pass) {

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_check_Supported_ISO_Namespace(in SupportedAppProtocolReq v_supportedAppProtocolReq,
                                             in verdicttype v_vct)
                                             runs on EVCC_Tester {

        for (var integer i:=0; i<lengthof(v_supportedAppProtocolReq.appProtocol_list); i:=i+1) {

            if(v_supportedAppProtocolReq.appProtocol_list[i].protocolNamespace ==
               vc_Supported_ISO_Namespace) {
               vc_SchemaID := v_supportedAppProtocolReq.appProtocol_list[i].schemaID;
               log("The requested schema is supported by the SECC.");
               return;
            }
        }
        setverdict(v_vct,"The requested schema is not supported by the SECC.");
        return;
    }
}
```

I.3.2 V2G messages

I.3.2.1 EVCC functions for SessionSetup

```
module TestBehavior_EVCC_SessionSetup {

    import from ComponentsAndPorts all;
    import from Timer_15118_2 all;
    import from Pics_15118 all;
    import from Services_HAL_61851_1 all;
    import from Templates_CMN_SupportedAppProtocol {
        template md_CMN_CMN_SupportedAppProtocolRes_001;
    }
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from Templates_CMN_SessionSetup all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_V2GMessage {
        template md_CMN_V2G_Message_001,
                 md_CMN_MessageHeaderType_001,
                 md_CMN_BodyType_001;
    }
    import from urn_iso_15118_2_2010_AppProtocol language "XSD" all;
    import from Templates_EVCC_SupportedAppProtocol all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from TestBehavior_EVCC_ServiceDiscovery all;
    import from DataStructure_HAL_61851 all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from LibFunctions_15118_2 all;
import from Timer_15118 all;

function f_EVCC_CMN_TB_VTB_SessionSetup_001(in verdicttype v_vct)
                                            runs on EVCC_Tester
                                            return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;
    var integer v_id;
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_CMN_SupportedAppProtocolRes_001(
                                oK_SuccessfulNegotiation, vc_SchemaID));
    alt {
        [not PICS_CMN_CMN_CombinedTesting] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                            md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(?),
                                            md_CMN_BodyType_001(
                                            md_CMN_CMN_SessionSetupReq_001(?))))
                                            -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            vc_SessionID := v_requestMessage.header.sessionID;
            if(f_is_SessionID_zero(vc_SessionID)){
                   setverdict(pass, "SessionSetupReq message was correct.");
            }
            else {
                    setverdict(v_vct,"SessionSetup message was not correct. " &
                                     "Received sessionID " &
                                    "was not equal to zero.");
            }
        }
        [PICS_CMN_CMN_CombinedTesting] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                            md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(?),
                                            md_CMN_BodyType_001(
                                            md_CMN_CMN_SessionSetupReq_001(
                                            hex2oct(vc_sut_mac)))))
                                            -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            vc_SessionID := v_requestMessage.header.sessionID;
            if(f_is_SessionID_zero(vc_SessionID)){
                setverdict(pass, "SessionSetupReq message was correct.");
            }
            else {
                    setverdict(v_vct,"SessionSetup message was not correct. " &
                                    "Received sessionID " &
                                    "was not equal to zero.");
            }
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {sessionSetupReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received SessionSetupReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_SessionSetup_002(urn_iso_15118_2_2013_MsgDataTypes.
                                            ResponseCodeType responseCode)
                                            runs on EVCC_Tester return verdicttype {

    var charstring v_message;

      f_EVCC_setPwmMode(e_PosVolt12);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                md_CMN_BodyType_001(
                                md_CMN_CMN_SessionSetupRes_001(
                                responseCode, vc_EvseId,
                                fx_getTimeAsLong()))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_ServiceDiscoveryReq_003(*,*)))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received ServiceDiscoveryReq message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }

    return getverdict;
}

function f_EVCC_CMN_TB_VTB_SessionSetup_003() runs on EVCC_Tester return verdicttype  {

    timer t_wait;

    // Wait until SUT message timeout
    t_wait.start(par_V2G_EVCC_Msg_Timeout_SessionSetupReq -
                 par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_SessionSetup_004() runs on EVCC_Tester return verdicttype {

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_SessionSetup_005(HAL_61851_PwmMode_Type pwmMode)
                                            runs on EVCC_Tester
                                            return verdicttype  {
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        vc_SessionID := f_rnd_SessionID(1,429496);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     md_CMN_CMN_SessionSetupRes_001(
                                     oK_NewSessionEstablished, vc_EvseId,
                                     fx_getTimeAsLong()))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                                                    md_CMN_BodyType_001(
                                                    md_CMN_CMN_ServiceDiscoveryReq_003(*,*)))) {
                    setverdict(fail,"Received ServiceDiscoveryReq message was not expected. " &
                                        "CP State E/F should be detected before.");
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                    setverdict(fail,"SUT has not terminated the TCP connection " &
                                    "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_is_SessionID_zero(SessionIDType sessionID) return boolean {

        if(oct2int(sessionID)==0){
            return true;
        }
        else {
            return false;
        }
    }

    function f_EVCC_CMN_TB_VTB_SessionSetup_006(in verdicttype v_vct)
                                                runs on EVCC_Tester
                                                return verdicttype {

        var V2G_Message v_requestMessage;
        var charstring v_message;
        var integer v_id;
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_CMN_SupportedAppProtocolRes_001(
                                        oK_SuccessfulNegotiation, vc_SchemaID));
        alt {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_SessionSetupReq_001(?)))) {

                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass,"SessionSetupRes message with previous sessionID " &
                                "was correct after paused session.");
            }

            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                           "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {sessionSetupReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received SessionSetupReq message, " &
                                                        "but with invalid data content.");
            [] a_EVCC_Timeout(v_vct);
        }
        return getverdict;
    }
}
```

I.3.2.2 EVCC functions for ServiceDiscovery

```
module TestBehavior_EVCC_ServiceDiscovery {

    import from ComponentsAndPorts all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_CommonBehavior all;
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Timer_15118_2 all;
import from Services_HAL_61851_1 all;
import from Pics_15118 all;
import from Pics_15118_2 all;
import from Templates_CMN_ServiceDiscovery all;
import from Templates_CMN_SessionSetup all;
import from Templates_EVCC_ServiceDetail all;
import from Templates_CMN_V2GMessage all;
import from Templates_EVCC_PaymentServiceSelection all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from DataStructure_HAL_61851 all;
import from LibFunctions_15118_2 all;
import from Timer_15118 all;

function f_EVCC_CMN_TB_VTB_ServiceDiscovery_001(in verdicttype v_vct)
                                                runs on EVCC_Tester
                                                return verdicttype {

    var charstring v_message;
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    vc_SessionID := f_rnd_SessionID(1,429496);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_CMN_SessionSetupRes_001(
                                  oK_NewSessionEstablished, vc_EvseId,
                                  fx_getTimeAsLong()))));

    alt {
        []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_ServiceDiscoveryReq_003(*,*)))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(pass,"ServiceDiscoveryReq message was correct.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                      {serviceDiscoveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received ServiceDiscoveryReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ServiceDiscovery_002(urn_iso_15118_2_2013_MsgDataTypes.
                                                ResponseCodeType responseCode)
                                                runs on EVCC_Tester
                                                return verdicttype  {

    var charstring v_message;

      f_EVCC_setPwmMode(e_PosVolt12);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_CMN_ServiceDiscoveryRes_002(
                                  responseCode, externalPayment,
                                  vc_FreeService,
                                  vc_requestedEnergyTransferMode,
                                  vc_ServiceId))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                        ?, ?)))) {

            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received PaymentServiceSelectionReq message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ServiceDiscovery_003() runs on EVCC_Tester return verdicttype  {

    timer t_wait;

    // Wait until SUT message timeout
    t_wait.start(par_V2G_EVCC_Msg_Timeout_ServiceDiscoveryReq -
                 par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ServiceDiscovery_004(HAL_61851_PwmMode_Type pwmMode)
                                            runs on EVCC_Tester
                                            return verdicttype {

    var ServiceListType v_serviceList;
    var boolean v_tcpTerminated := false;
    timer t_wait;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    if(PICS_CMN_CMN_IdentificationMode == pnC){
        vc_SelectedPaymentOption := contract;
        v_serviceList := m_CMN_CMN_ServiceList_001();
    } else {
        vc_SelectedPaymentOption := externalPayment;
        v_serviceList := m_CMN_CMN_ServiceList_002();
    }

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryRes_001(
                                    vc_SelectedPaymentOption,
                                    vc_ServiceId, vc_FreeService,
                                    vc_requestedEnergyTransferMode,
                                    v_serviceList))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ServiceDetailReq_001(?)))) {
               setverdict(fail,"Received ServiceDetailReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                          ?, ?)))) {
               setverdict(fail,"Received PaymentServiceSelectionReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}


function f_EVCC_CMN_TB_VTB_ServiceDiscovery_005() runs on EVCC_Tester return verdicttype {

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_ServiceDiscovery_006(in verdicttype v_vct)
                                                   runs on EVCC_Tester
                                                   return verdicttype {

        var charstring v_message;
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_SessionSetupRes_001(
                                    oK_OldSessionJoined, vc_EvseId,
                                    fx_getTimeAsLong()))));

        alt {
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_ServiceDiscoveryReq_003(*,*)))) {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass,"ServiceDiscoveryReq message was correct.");
            }

            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                           "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                         {serviceDiscoveryReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                          "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received ServiceDiscoveryReq message, " &
                                                        "but with invalid data content.");
            [] a_EVCC_Timeout(v_vct);
        }
        return getverdict;
    }
}
```

I.3.2.3 EVCC functions for ServiceDetail

```
module TestBehavior_EVCC_ServiceDetail {

    import from ComponentsAndPorts all;
    import from Pics_15118_2 all;
    import from Pics_15118 all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TestBehavior_EVCC_CommonBehavior all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Timer_15118_2 all;
import from Templates_EVCC_ServiceDetail all;
import from Templates_CMN_ServiceDiscovery all;
import from Templates_CMN_V2GMessage all;
import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
import from Templates_EVCC_ServiceDetail all;
import from Services_HAL_61851_1 all;
import from Templates_EVCC_PaymentServiceSelection all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from DataStructure_HAL_61851 all;
import from LibFunctions_15118_2 all;
import from Timer_15118 all;

//  PnC mode
function f_EVCC_CMN_TB_VTB_ServiceDetail_001(in verdicttype v_vct)
                                             runs on EVCC_Tester
                                             return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;
    var ServiceListType v_serviceList := m_CMN_CMN_ServiceList_001();
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    // Force the SUT to use Pnc mode by providing just one PaymentOption == contract//
    if(PICS_CMN_CMN_IdentificationMode == pnC){
        vc_SelectedPaymentOption := contract;
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryRes_001(
                                    vc_SelectedPaymentOption, vc_ServiceId,
                                    vc_FreeService,
                                    vc_requestedEnergyTransferMode,
                                    v_serviceList))));
    }else {
        setverdict(inconc, "Check IdentificationMode");
    }

    alt{
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_EVCC_CMN_ServiceDetailReq_001(?))))
                                         -> value v_requestMessage {

            vc_ServiceId := v_requestMessage.body.bodyElement.serviceDetailReq.serviceID;
            if(f_checkServiceID(vc_ServiceId, v_serviceList, v_vct)) {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass, "ServiceDetailReq message was correct.");
            }
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                     {serviceDetailReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received ServiceDetailReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

//  EIM mode
function f_EVCC_CMN_TB_VTB_ServiceDetail_002(in verdicttype v_vct)
                                             runs on EVCC_Tester
                                             return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;
    var ServiceListType v_serviceList := m_CMN_CMN_ServiceList_002();
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    // Force the SUT to use EIM mode by providing just one PaymentOption == external payment//
    if(PICS_CMN_CMN_IdentificationMode == eIM){
        vc_SelectedPaymentOption := externalPayment;
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryRes_001(
                                    vc_SelectedPaymentOption,
                                    vc_ServiceId, vc_FreeService,
                                    vc_requestedEnergyTransferMode,
                                    v_serviceList))));
    }else {
        setverdict(inconc, "Check IdentificationMode");
    }

    alt{
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_ServiceDetailReq_001(?))))
                                        -> value v_requestMessage {

            vc_ServiceId := v_requestMessage.body.bodyElement.serviceDetailReq.serviceID;
            if(f_checkServiceID(vc_ServiceId, v_serviceList, v_vct)) {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass, "ServiceDetailReq message was correct.");
            }
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                    {serviceDetailReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                              "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received ServiceDetailReq message, " &
                                             "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ServiceDetail_003(urn_iso_15118_2_2013_MsgDataTypes.
                                        ResponseCodeType responseCode)
                                        runs on EVCC_Tester return verdicttype {

    var charstring v_message;

      f_EVCC_setPwmMode(e_PosVolt12);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_ServiceDetailRes_001(
                                responseCode, vc_ServiceId))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                        ?, ?)))) {

            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received PaymentServiceSelectionReq message was not expected.");
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ServiceDetail_004() runs on EVCC_Tester return verdicttype  {

    timer t_wait;

    // Wait until SUT message timeout
    t_wait.start(par_V2G_EVCC_Msg_Timeout_ServiceDetailReq -
                 par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ServiceDetail_005() runs on EVCC_Tester return verdicttype {

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ServiceDetail_006(HAL_61851_PwmMode_Type pwmMode)
                                             runs on EVCC_Tester
                                             return verdicttype  {
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                   md_CMN_BodyType_001(
                                   md_EVCC_CMN_ServiceDetailRes_001(
                                   oK,vc_ServiceId))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                         ?, ?)))) {

               setverdict(fail,"Received PaymentServiceSelectionReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_checkServiceID(ServiceIDType v_serviceID, ServiceListType vc_serviceList,
                          in verdicttype v_vct)
                          return boolean {

  for (var integer i:=0; i<sizeof(vc_serviceList.service_list); i:=i + 1) {
      if(v_serviceID == vc_serviceList.service_list[i].serviceType.serviceID){
          return true;
      }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
            setverdict(v_vct, "Invalid Service ID was chosen.");
            return false;

    }
}
```

I.3.2.4 EVCC functions for ServiceDetailAndPaymentSelection

```
module TestBehavior_EVCC_ServiceDetailAndPaymentSelection {

    import from ComponentsAndPorts all;
    import from Pics_15118_2 all;
    import from Pixit_15118_2 all;
    import from Pics_15118 all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Timer_15118_2 all;
    import from Templates_EVCC_ServiceDetail all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from Templates_EVCC_ServiceDetail all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Services_HAL_61851_1 all;
    import from Templates_CMN_Authorization all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from TestBehavior_EVCC_ServiceDetail all;
    import from Templates_CMN_Authorization all;
    import from DataStructure_HAL_61851 all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118 all;

    //  PnC mode
    function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_001(in verdicttype v_vct)
                                                                    runs on EVCC_Tester
                                                                    return verdicttype {
        var V2G_Message v_requestMessage;
        var charstring v_message;
        var ServiceListType v_serviceList := m_CMN_CMN_ServiceList_001();
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

        // Force the SUT to use Pnc mode by providing just one PaymentOption == contract//
        if(PICS_CMN_CMN_IdentificationMode == pnC){
            vc_SelectedPaymentOption := contract;
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_ServiceDiscoveryRes_001(
                                        vc_SelectedPaymentOption,
                                        vc_ServiceId, vc_FreeService,
                                        vc_requestedEnergyTransferMode,
                                        v_serviceList))));
        }else {
            setverdict(inconc, "ServiceDetailAndPaymentSelection: Check IdentificationMode");
        }

        alt {
            //  Testbehavior for PaymentServiceSelection
                // ServiceList contains one entry with serviceID == vc_serviceID
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                            contract, ?)))) -> value v_requestMessage {
                tc_V2G_SECC_Sequence_Timer.stop;
                vc_selectedServiceList := v_requestMessage.body.bodyElement.
                                        paymentServiceSelectionReq.
                                        selectedServiceList;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                setverdict(pass, "PaymentServiceSelectionReq message was correct.");
            }
            // Testbehavior for ServiceDetail
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ServiceDetailReq_001(?))))
                                                -> value v_requestMessage {

                vc_ServiceId := v_requestMessage.body.bodyElement.serviceDetailReq.serviceID;
                if(f_checkServiceID(vc_ServiceId, v_serviceList, v_vct)) {
                    tc_V2G_SECC_Sequence_Timer.stop;
                    tc_V2G_SECC_Sequence_Timer.start;
                        if(vc_ServiceId == 2) {
                            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                                        md_CMN_BodyType_001(
                                                        md_EVCC_CMN_ServiceDetailRes_001(
                                                        oK,vc_ServiceId))));
                        } else if(vc_ServiceId == 3) {
                            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                                        md_CMN_BodyType_001(
                                                        md_EVCC_CMN_ServiceDetailRes_002(
                                                        oK,vc_ServiceId))));
                        }
                    setverdict(pass, "ServiceDetailAndPaymentSelection: ServiceDetailReq message " &
                                     "was correct. Waiting for PaymentServiceSelectionReq message.");
                    repeat;
                }
            }
            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                           "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                       {serviceDetailReq := ?},
                                                       {paymentServiceSelectionReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PaymentServiceSelectionReq or " &
                                                        "ServiceDetailReq message, " &
                                                        "but with invalid " &
                                                        "data content.");
            [] a_EVCC_Timeout(v_vct);
        }
        return getverdict;
    }


    //  EIM mode
    function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_002(in verdicttype v_vct)
                                                                    runs on EVCC_Tester
                                                                    return verdicttype {

        var V2G_Message v_requestMessage;
        var charstring v_message;
        var ServiceListType v_serviceList := m_CMN_CMN_ServiceList_002();
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

        // Force the SUT to use EIM mode by providing just one PaymentOption == externalPayment//
        if(PICS_CMN_CMN_IdentificationMode == eIM){
            vc_SelectedPaymentOption := externalPayment;
            if(vc_Security == cc_hexTls){
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_CMN_CMN_ServiceDiscoveryRes_001(
                                            vc_SelectedPaymentOption,
                                            vc_ServiceId, vc_FreeService,
                                            vc_requestedEnergyTransferMode,
                                            v_serviceList))));
            }
            else {
                // no serviceList (no VAS)
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_ServiceDiscoveryRes_002(
                                        oK, vc_SelectedPaymentOption,
                                        vc_FreeService,
                                        vc_requestedEnergyTransferMode,
                                        vc_ServiceId))));
            }
        }else {
            setverdict(inconc, "ServiceDetailAndPaymentSelection: " &
                               "Check PICS_CMN_CMN_IdentificationMode");
        }

        alt {
            //  Testbehavior for PaymentServiceSelection
                // ServiceList contains one entry with serviceID == vc_serviceID
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                          externalPayment, ?))))
                                          -> value v_requestMessage {
                tc_V2G_SECC_Sequence_Timer.stop;
                vc_selectedServiceList := v_requestMessage.body.bodyElement.
                                          paymentServiceSelectionReq.
                                          selectedServiceList;
                setverdict(pass, "PaymentServiceSelectionReq message was correct.");
            }
            // Testbehavior for ServiceDetail
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ServiceDetailReq_001(?))))
                                          -> value v_requestMessage {

                vc_ServiceId := v_requestMessage.body.bodyElement.serviceDetailReq.serviceID;
                if(f_checkServiceID(vc_ServiceId, v_serviceList, v_vct)) {
                    tc_V2G_SECC_Sequence_Timer.stop;
                    tc_V2G_SECC_Sequence_Timer.start;
                        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  md_EVCC_CMN_ServiceDetailRes_002(
                                                  oK,vc_ServiceId))));
                    setverdict(pass, "ServiceDetailAndPaymentSelection: ServiceDetailReq message " &
                                     "was correct. Waiting for PaymentServiceSelectionReq message.");
                    repeat;
                }
            }
            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                           "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                       {serviceDetailReq := ?},
                                                       {paymentServiceSelectionReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PaymentServiceSelectionReq or " &
                                                        "ServiceDetailReq message, " &
                                                        "but with invalid " &
                                                        "data content.");
            [] a_EVCC_Timeout(v_vct);
        }
        return getverdict;
    }

    //  PnC mode
    function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_003() runs on EVCC_Tester
                                                                      return verdicttype {

        var V2G_Message v_requestMessage;
        var boolean v_serviceDetailReceived := false;

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        var ServiceListType v_serviceList := m_CMN_CMN_ServiceList_001();

        // Force the SUT to use Pnc mode by providing just one PaymentOption == contract//
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
if(PICS_CMN_CMN_IdentificationMode == pnC){
    vc_SelectedPaymentOption := contract;
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryRes_001(
                                vc_SelectedPaymentOption, vc_ServiceId,
                                vc_FreeService, vc_requestedEnergyTransferMode,
                                v_serviceList))));

}else {
    setverdict(inconc, "ServiceDetailAndPaymentSelection: Check IdentificationMode");
}

alt    {
    //  Testbehavior for PaymentServiceSelection
    [v_serviceDetailReceived] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                            md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(
                                            vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                            contract, ?))))
                                            -> value v_requestMessage {
        tc_V2G_SECC_Sequence_Timer.stop;
        var SelectedServiceListType v_selectedServiceList := v_requestMessage.
                                                             body.bodyElement.
                                                             paymentServiceSelectionReq.
                                                             selectedServiceList;
        f_checkChosenPaymentService(v_selectedServiceList);
        if(getverdict == pass) {
            setverdict(pass, "PaymentServiceSelectionReq message was correct.");
        }
    }
    // Testbehavior for ServiceDetail
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(
                                        vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_ServiceDetailReq_001(?))))
                                        -> value v_requestMessage {

        vc_ServiceId := v_requestMessage.body.bodyElement.serviceDetailReq.serviceID;
        if(f_checkServiceID(vc_ServiceId, v_serviceList, fail)) {
            tc_V2G_SECC_Sequence_Timer.stop;
            tc_V2G_SECC_Sequence_Timer.start;
                if(vc_ServiceId == 2) {
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ServiceDetailRes_001(
                                                oK,vc_ServiceId))));
                } else if(vc_ServiceId == 3) {
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ServiceDetailRes_002(
                                                oK,vc_ServiceId))));
                }
            setverdict(pass, "ServiceDetailAndPaymentSelection: ServiceDetailReq message " &
                             "was correct. Waiting for PaymentServiceSelectionReq message.");
            v_serviceDetailReceived := true;
            repeat;
        }
    }
    [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                  "terminated by the SUT.");
    [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(fail, "Unexpected V2G message was received.",
                                            {serviceDetailReq := ?},
                                            {paymentServiceSelectionReq := ?});
    [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                             "but cannot be decoded.");
    [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionReq or " &
                                            "ServiceDetailReq message, " &
                                            "but with invalid " &
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                    "data content.");
            [] a_EVCC_Timeout(fail);
        }
        return getverdict;
    }

    //  EIM mode
    function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_004() runs on EVCC_Tester
                                                                      return verdicttype {
        var V2G_Message v_requestMessage;
        var boolean v_serviceDetailReceived := false;

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        var ServiceListType v_serviceList := m_CMN_CMN_ServiceList_002();

        // Force the SUT to use EIM mode by providing just one PaymentOption == externalPayment//
        if(PICS_CMN_CMN_IdentificationMode == eIM){
            vc_SelectedPaymentOption := externalPayment;
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_ServiceDiscoveryRes_001(
                                        vc_SelectedPaymentOption,
                                        vc_ServiceId, vc_FreeService,
                                        vc_requestedEnergyTransferMode,
                                        v_serviceList))));
        }else {
            setverdict(inconc, "ServiceDetailAndPaymentSelection: " &
                               "Check PICS_CMN_CMN_IdentificationMode");
        }

        alt
        {
            //  Testbehavior for PaymentServiceSelection
            [v_serviceDetailReceived] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                                  md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(
                                                  vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                                  externalPayment, ?))))
                                                  -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                var SelectedServiceListType v_selectedServiceList := v_requestMessage.
                                                                     body.bodyElement.
                                                                     paymentServiceSelectionReq.
                                                                     selectedServiceList;
                f_checkChosenPaymentService(v_selectedServiceList);
                if(getverdict == pass) {
                    setverdict(pass, "PaymentServiceSelectionReq message was correct.");
                }
            }
            // Testbehavior for ServiceDetail
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(
                                                 vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_EVCC_CMN_ServiceDetailReq_001(?))))
                                                 -> value v_requestMessage {

                vc_ServiceId := v_requestMessage.body.bodyElement.serviceDetailReq.serviceID;
                if(f_checkServiceID(vc_ServiceId, v_serviceList, fail)) {
                    tc_V2G_SECC_Sequence_Timer.stop;
                    tc_V2G_SECC_Sequence_Timer.start;
                        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                                     md_CMN_BodyType_001(
                                                     md_EVCC_CMN_ServiceDetailRes_002(
                                                     oK,vc_ServiceId))));
                    setverdict(pass, "ServiceDetailAndPaymentSelection: ServiceDetailReq message " &
                                     "was correct. Waiting for PaymentServiceSelectionReq message.");
                    v_serviceDetailReceived := true;
                    repeat;
                }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(fail, "Unexpected V2G message was received.",
                                                   {serviceDetailReq := ?},
                                                   {paymentServiceSelectionReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionReq or " &
                                                   "ServiceDetailReq message, " &
                                                   "but with invalid " &
                                                   "data content.");
        [] a_EVCC_Timeout(fail);
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_005(
                                                        urn_iso_15118_2_2013_MsgDataTypes.
                                                        ResponseCodeType responseCode)
                                                        runs on EVCC_Tester
                                                        return verdicttype  {

    var charstring v_message;

      f_EVCC_setPwmMode(e_PosVolt12);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_EVCC_CMN_PaymentServiceSelectionRes_001(
                                 responseCode))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_AuthorizationRes_001(?,?)))) {

            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received AuthorizationRes message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }

    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_006() runs on EVCC_Tester
                                                                  return verdicttype {

    timer t_wait;

    // Wait until SUT message timeout
    t_wait.start(par_V2G_EVCC_Msg_Timeout_PaymentServiceSelectionReq -
                 par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_007() runs on EVCC_Tester
                                                                  return verdicttype {

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_AuthorizationRes_001(oK, finished))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                   "but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_008(HAL_61851_PwmMode_Type pwmMode)
                                                                 runs on EVCC_Tester
                                                                 return verdicttype  {
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT " &
                                                          "before request message " &
                                                          "was sent by the Test System. " &
                                                          "The send operation will be " &
                                                          "canceled.") {
                v_tcpTerminated := true;
            }
        }
    if(not v_tcpTerminated) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                           "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT") {
               log("CP State E/F should be detected before.");
            }
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

//  PnC mode
function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_009() runs on EVCC_Tester
                                                                  return verdicttype {

    var V2G_Message v_requestMessage;
    var boolean v_serviceDetailReceived := false;

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    var ServiceListType v_serviceList := m_CMN_CMN_ServiceList_001();

    // Force the SUT to use Pnc mode by providing just one PaymentOption == contract//
    if(PICS_CMN_CMN_IdentificationMode == pnC){
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryRes_004(
                                    contract, vc_ServiceId, vc_FreeService,
                                    vc_requestedEnergyTransferMode,
                                    v_serviceList))));

    }else {
        setverdict(inconc, "ServiceDetailAndPaymentSelection: Check IdentificationMode");
    }

    alt    {
        //  Testbehavior for PaymentServiceSelection
        [v_serviceDetailReceived] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                          md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(
                                          vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                          contract, ?))))
                                          -> value v_requestMessage {
            tc_V2G_SECC_Sequence_Timer.stop;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            var SelectedServiceListType v_selectedServiceList := v_requestMessage.
                                                                 body.bodyElement.
                                                                 paymentServiceSelectionReq.
                                                                 selectedServiceList;
            f_checkChosenPaymentService(v_selectedServiceList);
            if(getverdict == pass) {
                setverdict(pass, "PaymentServiceSelectionReq message was correct. " &
                                 "The previous SelectedPaymentOption was chosen after " &
                                 "paused session.");
            }
        }
        // Testbehavior for ServiceDetail
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(
                                                vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ServiceDetailReq_001(?))))
                                                -> value v_requestMessage {

            vc_ServiceId := v_requestMessage.body.bodyElement.serviceDetailReq.serviceID;
            if(f_checkServiceID(vc_ServiceId, v_serviceList, fail)) {
                tc_V2G_SECC_Sequence_Timer.stop;
                tc_V2G_SECC_Sequence_Timer.start;
                    if(vc_ServiceId == 2) {
                        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                                          md_CMN_BodyType_001(
                                                          md_EVCC_CMN_ServiceDetailRes_001(
                                                          oK,vc_ServiceId))));
                    } else if(vc_ServiceId == 3) {
                        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                                          md_CMN_BodyType_001(
                                                          md_EVCC_CMN_ServiceDetailRes_002(
                                                          oK,vc_ServiceId))));
                    }
                setverdict(pass, "ServiceDetailAndPaymentSelection: ServiceDetailReq message " &
                                 "was correct. Waiting for PaymentServiceSelectionReq message.");
                v_serviceDetailReceived := true;
                repeat;
            }
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(fail, "Unexpected V2G message was received.",
                                                    {serviceDetailReq := ?},
                                                    {paymentServiceSelectionReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionReq or " &
                                                   "ServiceDetailReq message, " &
                                                   "but with invalid " &
                                                   "data content.");
        [] a_EVCC_Timeout(fail);
    }
    return getverdict;
}

//  EIM mode
function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_010() runs on EVCC_Tester
                                                                  return verdicttype {
    var V2G_Message v_requestMessage;
    var boolean v_serviceDetailReceived := false;

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    var ServiceListType v_serviceList := m_CMN_CMN_ServiceList_002();

    // Force the SUT to use EIM mode by providing just one PaymentOption == externalPayment//
    if(PICS_CMN_CMN_IdentificationMode == eIM){
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_ServiceDiscoveryRes_004(
                                          externalPayment, vc_ServiceId, vc_FreeService,
                                          vc_requestedEnergyTransferMode,
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    v_serviceList))));
    }else {
        setverdict(inconc, "ServiceDetailAndPaymentSelection: " &
                           "Check PICS_CMN_CMN_IdentificationMode");
    }

    alt {
        //  Testbehavior for PaymentServiceSelection
        [v_serviceDetailReceived] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                          md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(
                                          vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                          externalPayment, ?)))) {

            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(pass, "PaymentServiceSelectionReq message was correct. " &
                             "The previous SelectedPaymentOption was chosen after " &
                             "paused session.");
        }
        // Testbehavior for ServiceDetail
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(
                                            vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_EVCC_CMN_ServiceDetailReq_001(?))))
                                            -> value v_requestMessage {

            vc_ServiceId := v_requestMessage.body.bodyElement.serviceDetailReq.serviceID;
            if(f_checkServiceID(vc_ServiceId, v_serviceList, fail)) {
                tc_V2G_SECC_Sequence_Timer.stop;
                tc_V2G_SECC_Sequence_Timer.start;
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  md_EVCC_CMN_ServiceDetailRes_002(
                                                  oK,vc_ServiceId))));
                setverdict(pass, "ServiceDetailAndPaymentSelection: ServiceDetailReq message " &
                                 "was correct. Waiting for PaymentServiceSelectionReq message.");
                v_serviceDetailReceived := true;
                repeat;
            }
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(fail, "Unexpected V2G message was received.",
                                                  {serviceDetailReq := ?},
                                                  {paymentServiceSelectionReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionReq or " &
                                                   "ServiceDetailReq message, " &
                                                   "but with invalid " &
                                                   "data content.");
        [] a_EVCC_Timeout(fail);
    }
    return getverdict;
}

//  PnC mode
function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_011(in verdicttype v_vct)
                                                                  runs on EVCC_Tester
                                                                  return verdicttype {
    var V2G_Message v_requestMessage;
    var charstring v_message;
    var ServiceListType v_serviceList := m_CMN_CMN_ServiceList_001();
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    // Force the SUT to use Pnc mode by providing just one PaymentOption == contract//
    if(PICS_CMN_CMN_IdentificationMode == pnC){
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryRes_004(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    contract, vc_ServiceId, vc_FreeService,
                                    vc_requestedEnergyTransferMode,
                                    v_serviceList))));
    }else {
        setverdict(inconc, "ServiceDetailAndPaymentSelection: Check IdentificationMode");
    }

    alt {
        //  Testbehavior for PaymentServiceSelection
            // ServiceList contains one entry with serviceID == vc_serviceID
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                        contract, ?)))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(pass, "PaymentServiceSelectionReq message was correct. " &
                             "The previous SelectedPaymentOption was chosen after " &
                             "paused session.");
        }
        // Testbehavior for ServiceDetail
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_ServiceDetailReq_001(?))))
                                        -> value v_requestMessage {

            vc_ServiceId := v_requestMessage.body.bodyElement.serviceDetailReq.serviceID;
            if(f_checkServiceID(vc_ServiceId, v_serviceList, v_vct)) {
                tc_V2G_SECC_Sequence_Timer.stop;
                tc_V2G_SECC_Sequence_Timer.start;
                    if(vc_ServiceId == 2) {
                        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ServiceDetailRes_001(
                                                oK,vc_ServiceId))));
                    } else if(vc_ServiceId == 3) {
                        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ServiceDetailRes_002(
                                                oK,vc_ServiceId))));
                    }
                setverdict(pass, "ServiceDetailAndPaymentSelection: ServiceDetailReq message " &
                                 "was correct. Waiting for PaymentServiceSelectionReq message.");
                repeat;
            }
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                   {serviceDetailReq := ?},
                                                   {paymentServiceSelectionReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PaymentServiceSelectionReq or " &
                                                    "ServiceDetailReq message, " &
                                                    "but with invalid " &
                                                    "data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

//  EIM mode
function f_EVCC_CMN_TB_VTB_ServiceDetailAndPaymentSelection_012(in verdicttype v_vct)
                                                                runs on EVCC_Tester
                                                                return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;
    var ServiceListType v_serviceList := m_CMN_CMN_ServiceList_002();
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
// Force the SUT to use EIM mode by providing just one PaymentOption == externalPayment//
if(PICS_CMN_CMN_IdentificationMode == eIM){
    if(vc_Security == cc_hexTls){
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_ServiceDiscoveryRes_004(
                                        externalPayment, vc_ServiceId, vc_FreeService,
                                        vc_requestedEnergyTransferMode,
                                        v_serviceList))));
    }
    else {
        // no serviceList (no VAS)
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_ServiceDiscoveryRes_004(
                                        externalPayment, vc_ServiceId, vc_FreeService,
                                        vc_requestedEnergyTransferMode,
                                        omit))));
    }
}else {
    setverdict(inconc, "ServiceDetailAndPaymentSelection: " &
                       "Check PICS_CMN_CMN_IdentificationMode");
}

alt {
    //  Testbehavior for PaymentServiceSelection
        // ServiceList contains one entry with serviceID == vc_serviceID
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                          externalPayment, ?)))) {
        tc_V2G_SECC_Sequence_Timer.stop;
        setverdict(pass, "PaymentServiceSelectionReq message was correct. " &
                         "The previous SelectedPaymentOption was chosen after " &
                         "paused session.");
    }
    // Testbehavior for ServiceDetail
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ServiceDetailReq_001(?))))
                                          -> value v_requestMessage {

        vc_ServiceId := v_requestMessage.body.bodyElement.serviceDetailReq.serviceID;
        if(f_checkServiceID(vc_ServiceId, v_serviceList, v_vct)) {
            tc_V2G_SECC_Sequence_Timer.stop;
            tc_V2G_SECC_Sequence_Timer.start;
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_EVCC_CMN_ServiceDetailRes_002(
                                                 oK,vc_ServiceId))));
            setverdict(pass, "ServiceDetailAndPaymentSelection: ServiceDetailReq message " &
                             "was correct. Waiting for PaymentServiceSelectionReq message.");
            repeat;
        }
    }

    [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                   "was misleadingly " &
                                                   "terminated by the SUT.");
    [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                               {serviceDetailReq := ?},
                                               {paymentServiceSelectionReq := ?});
    [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                 "but cannot be decoded.");
    [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PaymentServiceSelectionReq or " &
                                                "ServiceDetailReq message, " &
                                                "but with invalid " &
                                                "data content.");
    [] a_EVCC_Timeout(v_vct);
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        return getverdict;
    }

    function f_checkChosenPaymentService(SelectedServiceListType v_selectedServiceList) {

        for (var integer i:=0; i<sizeof(v_selectedServiceList.selectedService_list); i:=i + 1) {
            if(v_selectedServiceList.selectedService_list[i].serviceID == 2) {
                if(v_selectedServiceList.selectedService_list[i].parameterSetID < 1 or
                   v_selectedServiceList.selectedService_list[i].parameterSetID > 2) {
                     setverdict(fail,"Invalid parameterSetID was chosen.");
                }
                break;
            }
            else if(v_selectedServiceList.selectedService_list[i].serviceID == 3) {
                if(v_selectedServiceList.selectedService_list[i].parameterSetID < 1 or
                   v_selectedServiceList.selectedService_list[i].parameterSetID > 4) {
                     setverdict(fail,"Invalid parameterSetID was chosen.");
                }
            }
        }
    }
}
```

I.3.2.5 EVCC functions for PaymentServiceSelection

```
module TestBehavior_EVCC_PaymentServiceSelection {

    import from ComponentsAndPorts all;
    import from Pics_15118_2 all;
    import from Pics_15118 all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Timer_15118_2 all;
    import from Templates_CMN_ServiceDiscovery all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Services_HAL_61851_1 all;
    import from LibFunctions_15118_2 { group generalFunctions; }
    import from Timer_15118 all;

    //  PnC mode
    function f_EVCC_CMN_TB_VTB_PaymentServiceSelection_001() runs on EVCC_Tester
                                                             return verdicttype {

        var V2G_Message v_requestMessage;
        var charstring v_message;
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

        // Force the SUT to use Pnc mode by providing just one PaymentOption == contract//
        if(PICS_CMN_CMN_IdentificationMode == pnC){
            vc_SelectedPaymentOption := contract;
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_ServiceDiscoveryRes_001(
                                        vc_SelectedPaymentOption, vc_ServiceId,
                                        vc_FreeService, vc_requestedEnergyTransferMode,
                                        omit))));
        }else {
            setverdict(inconc, "ServiceDetailAndPaymentSelection: Check IdentificationMode");
        }

        alt {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                            contract, m_EVCC_CMN_SelectedServiceList_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        vc_ServiceId))))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(pass, "PaymentServiceSelectionReq message was correct.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {paymentServiceSelectionReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionReq " &
                                                   "message, but with invalid " &
                                                   "data content.");
        [] a_EVCC_Timeout(fail);
    }
    return getverdict;
}

//  EIM mode
function f_EVCC_CMN_TB_VTB_PaymentServiceSelection_002() runs on EVCC_Tester
                                                         return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    // Force the SUT to use EIM mode by providing just one PaymentOption == externalPayment//
    if(PICS_CMN_CMN_IdentificationMode == eIM){
        vc_SelectedPaymentOption := externalPayment;
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryRes_001(
                                    vc_SelectedPaymentOption, vc_ServiceId,
                                    vc_FreeService, vc_requestedEnergyTransferMode,
                                    omit))));
    }else {
        setverdict(inconc, "ServiceDetailAndPaymentSelection: " &
                           "Check PICS_CMN_CMN_IdentificationMode");
    }

    alt {
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                        externalPayment, m_EVCC_CMN_SelectedServiceList_001(
                                        vc_ServiceId))))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(pass, "PaymentServiceSelectionReq message was correct.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {paymentServiceSelectionReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionReq " &
                                                   "message, but with invalid " &
                                                   "data content.");
        [] a_EVCC_Timeout(fail);
    }
    return getverdict;
}

//  PnC mode
function f_EVCC_CMN_TB_VTB_PaymentServiceSelection_003() runs on EVCC_Tester
                                                         return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    // Force the SUT to use Pnc mode by providing just one PaymentOption == contract//
    if(PICS_CMN_CMN_IdentificationMode == pnC){
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryRes_004(
                                contract, vc_ServiceId, vc_FreeService,
                                vc_requestedEnergyTransferMode,
                                omit))));
    }else {
        setverdict(inconc, "ServiceDetailAndPaymentSelection: Check IdentificationMode");
    }

    alt {
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                    contract, m_EVCC_CMN_SelectedServiceList_001(
                                    vc_ServiceId))))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(pass, "PaymentServiceSelectionReq message was correct. " &
                             "The previous SelectedPaymentOption was chosen after " &
                             "paused session.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                {paymentServiceSelectionReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionReq " &
                                                   "message, but with invalid " &
                                                   "data content.");
        [] a_EVCC_Timeout(fail);
    }
    return getverdict;
}

//  EIM mode
function f_EVCC_CMN_TB_VTB_PaymentServiceSelection_004() runs on EVCC_Tester
                                                         return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    // Force the SUT to use EIM mode by providing just one PaymentOption == externalPayment//
    if(PICS_CMN_CMN_IdentificationMode == eIM){
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_ServiceDiscoveryRes_004(
                                externalPayment, vc_ServiceId,
                                vc_FreeService, vc_requestedEnergyTransferMode,
                                omit))));
    }else {
        setverdict(inconc, "ServiceDetailAndPaymentSelection: " &
                           "Check PICS_CMN_CMN_IdentificationMode");
    }

    alt {
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                    externalPayment, m_EVCC_CMN_SelectedServiceList_001(
                                    vc_ServiceId))))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(pass, "PaymentServiceSelectionReq message was correct. " &
                             "The previous SelectedPaymentOption was chosen after " &
                             "paused session.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                {paymentServiceSelectionReq := ?});
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionReq " &
                                                       "message, but with invalid " &
                                                       "data content.");
            [] a_EVCC_Timeout(fail);
        }
        return getverdict;
    }

    //  EIM mode
    function f_EVCC_AC_TB_VTB_PaymentServiceSelection_001(in float v_DutyCycle)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

        var V2G_Message v_requestMessage;
        var charstring v_message;
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

            f_EVCC_setPwmMode(e_PosVolt12);
            sleep(par_CMN_waitForNextHAL);
            f_EVCC_setDutyCycle(v_DutyCycle);
            sleep(par_CMN_waitForNextHAL);
            f_EVCC_setPwmMode(e_OscOn);


        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_ServiceDiscoveryRes_003(vc_ServiceId))));

        alt {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_EVCC_CMN_PaymentServiceSelectionReq_001(
                                            externalPayment, m_EVCC_CMN_SelectedServiceList_001(
                                            vc_ServiceId))))) {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass, "PaymentServiceSelectionReq message was correct. " &
                                 "The PaymentOptionType 'externalPayment' was " &
                                 "chosen by the SUT.");
            }
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                          "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                             {paymentServiceSelectionReq := ?});
            [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentServiceSelectionReq " &
                                                       "message, but with invalid " &
                                                       "data content.");
            [] a_EVCC_Timeout(fail);
        }
        return getverdict;
    }
}
```

I.3.2.6 EVCC functions for CertificateInstallation

```
module TestBehavior_EVCC_CertificateInstallation {

    import from Timer_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts {type EVCC_Tester;};
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Templates_EVCC_CertificateInstallation all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from Pics_15118 all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Services_HAL_61851_1 all;
import from Templates_EVCC_PaymentDetails all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from Services_Security all;
import from Templates_CMN_Security all;
import from urn_iso_15118_2_2013_X509Def all;
import from http_www_w3_org_2000_09_xmldsig all;
import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
import from LibFunctions_15118_2 all;
import from DataStructure_HAL_61851 all;
import from Timer_15118 all;
import from TTlibrary_Logging all;

function f_EVCC_CMN_TB_VTB_CertificateInstallation_001(in verdicttype v_vct)
                                                        runs on EVCC_Tester
                                                        return verdicttype {

    var charstring v_message;
    var V2G_Message v_requestMessage;
    var boolean isSignatureOk := true;
    var boolean v_isAuthentic := false;
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    var float v_processingDummy := 5.0;

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    alt {
         []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_003(vc_SessionID,
                                          md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                         md_CMN_BodyType_001(
                                         mdw_EVCC_CMN_CertificateInstallationReq_001(
                                         m_CMN_oEMProvisioningCertificate_001()))))
                                         -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
            vc_oEMProvisioningCert := v_requestMessage.body.bodyElement.
                                     certificateInstallationReq.oEMProvisioningCert;
            var Signature authSignature := v_requestMessage.header.signature_;

            var BodyType certificateInstallationReqBodyType := v_requestMessage.
                                                               body;

            f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                       cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

            var boolean v_result := fx_validateCertID(vc_oEMProvisioningCert.
                                                      tBSCertificate.subject);
            if(not v_result) {
                  setverdict(v_vct, "Invalid certID was detected within OEM provisioning " &
                                   "certificate.");
            }
            if(getverdict == pass) {
                    isSignatureOk := f_verifySignature(
                                     certificateInstallationReqBodyType,
                                     authSignature,
                                     vc_oEMProvisioningCert, v_vct);

                if(isSignatureOk) {
                      v_isAuthentic := fx_checkAuthOfOEMProvisioningCert(
                                                v_requestMessage);
                } else {
                   v_isAuthentic := true;
                }

                if(not v_isAuthentic) {
                          setverdict(v_vct,"Authentication of oEMProvisioningCert " &
                                          "was not correct.");
                }
                if(v_isAuthentic) {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                f_validateRootCertificateIDLength(certificateInstallationReqBodyType, v_vct);
                if(getverdict == pass) {
                    vc_sAProvisioningCertChain :=
                    fx_getSAProvisioningCertificateChainFromSA(vc_oEMProvisioningCert);
                    vc_contractSignatureCertChain := fx_getContractSignatureCertChainFromSA(
                                                      vc_oEMProvisioningCert);
                    vc_contractSignatureEncryptedPrivateKey :=
                    fx_getContractSignatureEncryptedPrivateKeyFromSA(vc_oEMProvisioningCert);
                    vc_dHpublickey := fx_getDHpublickeyFromSA(vc_oEMProvisioningCert);
                    vc_eMAID := fx_getEMAIDFromSA(vc_oEMProvisioningCert);

                    vc_myCertificateInstallationBodyType := md_CMN_BodyType_001(
                                                md_EVCC_CMN_CertificateInstallationRes_001(
                                                oK, vc_sAProvisioningCertChain,
                                                vc_contractSignatureCertChain,
                                                vc_contractSignatureEncryptedPrivateKey,
                                                vc_dHpublickey, vc_eMAID));

                    vc_signature := fx_getSignatureFromSA();

                    setverdict(pass,"CertificateInstallationReq message was correct.");
                    vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
                    tc_CMN_ProcessingTime_Timer.stop;
                }
            }
        }
    }
    [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                   "terminated by the SUT.");
    [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {certificateInstallationReq := ?});
    [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                  "but cannot be decoded.");
    [] a_EVCC_Unexpected_Message_Content(v_vct, "Received CertificateInstallationReq " &
                                                "message, but with invalid " &
                                                "data content.");
    [] a_EVCC_Timeout(v_vct);
    }
  return getverdict;
}

function f_EVCC_CMN_TB_VTB_CertificateInstallation_002(urn_iso_15118_2_2013_MsgDataTypes.
                                                   ResponseCodeType responseCode)
                                                   runs on EVCC_Tester return verdicttype  {

  var charstring v_message;

    f_EVCC_setPwmMode(e_PosVolt12);

  pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                              md_CMN_MessageHeaderType_001(vc_SessionID),
                              md_CMN_BodyType_001(
                              md_EVCC_CMN_CertificateInstallationRes_001(
                              responseCode, md_CertificateChainType_001(
                              vc_sAProvisioningCertChain.x509Certificate),
                              md_CertificateChainType_001(
                              vc_contractSignatureCertChain.x509Certificate),
                              vc_contractSignatureEncryptedPrivateKey,
                              vc_dHpublickey, vc_eMAID))));

  tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

  alt {
    [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                  "terminated by the SUT.");
    [] a_EVCC_CMN_SessionStop_Termination() {
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);
        repeat;
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentDetailsReq_001(?,?)))) {
        tc_V2G_SECC_Sequence_Timer.stop;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                setverdict(fail,"Received PaymentDetailsReq message was not expected.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_CertificateInstallation_003() runs on EVCC_Tester
                                                             return verdicttype {

        timer t_wait;

        // Wait until SUT message timeout
        log("Proccessing time: " & toString<float>(vc_processingTime) & "s.");
        t_wait.start(par_V2G_EVCC_Msg_Timeout_CertificateInstallationReq -
                     par_CMN_Transmission_Delay - vc_processingTime);
        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));

        alt {
            [] t_wait.timeout;
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                          "terminated by the SUT " &
                                                          "before timeout has expired.");
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
                setverdict(fail, "No V2G message was expected.");
            }
        }

        if(getverdict == pass) {


tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_CertificateInstallation_004() runs on EVCC_Tester return verdicttype {

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                   "but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_CertificateInstallation_005(HAL_61851_PwmMode_Type pwmMode)
                                                       runs on EVCC_Tester
                                                       return verdicttype  {
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }

    if(not v_tcpTerminated) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_003(vc_SessionID,
                                                               vc_signature),
                                  md_CMN_BodyType_001(
                                  md_EVCC_CMN_CertificateInstallationRes_001(
                                  oK, vc_sAProvisioningCertChain,
                                  vc_contractSignatureCertChain,
                                  vc_contractSignatureEncryptedPrivateKey,
                                  vc_dHpublickey, vc_eMAID))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                       md_CMN_MessageHeaderType_001(vc_SessionID),
                                       md_CMN_BodyType_001(
                                       md_EVCC_CMN_PaymentDetailsReq_001(
                                       ?,m_CMN_contractSignatureCertChain_001()))))  {

               setverdict(fail,"Received PaymentDetailsReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }
}
```

I.3.2.7 EVCC functions for CertificateUpdate

```
module TestBehavior_EVCC_CertificateUpdate {

    import from Timer_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts {type EVCC_Tester;};
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from Templates_EVCC_CertificateUpdate all;
    import from Pics_15118 all;
    import from Services_HAL_61851_1 all;
    import from Templates_EVCC_PaymentDetails all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Services_Security all;
    import from Templates_CMN_Security all;
    import from urn_iso_15118_2_2013_X509Def all;
    import from http_www_w3_org_2000_09_xmldsig all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from TTlibrary_Logging all;
    import from Timer_15118 all;

    function f_EVCC_CMN_TB_VTB_CertificateUpdate_001(in verdicttype v_vct)
                                                runs on EVCC_Tester
                                                return verdicttype {

        var charstring v_message;
        var V2G_Message v_requestMessage;
        var boolean isSignatureOk := true;
        var boolean v_isAuthentic := false;
        var float v_processingDummy := 5.0;
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

        alt {
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_003(vc_SessionID,
                                        md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                        md_CMN_BodyType_001(
                                        mdw_EVCC_CMN_CertificateUpdateReq_001(
                                        m_CMN_contractSignatureCertChain_001()))))
                                        -> value v_requestMessage {
                tc_V2G_SECC_Sequence_Timer.stop;
                tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
                vc_contractSignatureCert := v_requestMessage.body.bodyElement.
                                            certificateUpdateReq.
                                         contractSignatureCertChain.x509Certificate;
                vc_contractSignatureCertChain := v_requestMessage.body.
                                                 bodyElement.certificateUpdateReq.
                                              contractSignatureCertChain;
                var EMAIDType_1 v_eMAID := v_requestMessage.body.bodyElement.
                                        certificateUpdateReq.eMAID;
                var Signature authSignature := v_requestMessage.header.signature_;
                var BodyType certificateUpdateReqBodyType := v_requestMessage.
                                                             body;

                f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                         cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

                if(getverdict == pass) {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            isSignatureOk := f_verifySignature(
                             certificateUpdateReqBodyType,
                             authSignature,
                             vc_contractSignatureCert, v_vct);

        var boolean result := fx_checkCertificateValidity(vc_contractSignatureCertChain.
                                                          x509Certificate.tBSCertificate.
                                                          validity.not_Before,
                                                          vc_contractSignatureCertChain.
                                                          x509Certificate.tBSCertificate.
                                                          validity.not_After);
        if(not result) {
                setverdict(v_vct,"Invalid Certificate validity was detected.");
        }
        if(getverdict == pass) {
            log("EMAID: " & toString<charstring>(v_eMAID));
            result := fx_validateEMAID(v_eMAID);
            if(not result) {
                      setverdict(v_vct,"Invalid EMAID syntax was detected.");
            }
            if(isSignatureOk and getverdict == pass) {
               v_isAuthentic :=  fx_updateContractSignatureCertChain(
                                 v_requestMessage);
            } else {
               v_isAuthentic := true;
            }

            if(not v_isAuthentic) {
                 setverdict(v_vct,"Update of contractSignatureCertChain " &
                                  "was not successful.");
            }
            if(v_isAuthentic) {

                f_validateRootCertificateIDLength(certificateUpdateReqBodyType, v_vct);
             if(getverdict == pass) {

                 vc_sAProvisioningCertChain :=
                 fx_getSAProvisioningCertificateChainFromSA(vc_contractSignatureCert);
                 vc_contractSignatureCertChain :=
                 fx_getContractSignatureCertChainFromSA(vc_contractSignatureCert);
                 vc_contractSignatureEncryptedPrivateKey :=
                 fx_getContractSignatureEncryptedPrivateKeyFromSA(
                 vc_contractSignatureCert);
                 vc_dHpublickey := fx_getDHpublickeyFromSA(vc_contractSignatureCert);

                 vc_eMAID := fx_getEMAIDFromSA(vc_contractSignatureCert);

                 vc_myCertificateUpdateBodyType := md_CMN_BodyType_001(
                                              md_EVCC_CMN_CertificateUpdateRes_001(
                                              oK, vc_sAProvisioningCertChain,
                                              vc_contractSignatureCertChain,
                                              vc_contractSignatureEncryptedPrivateKey,
                                              vc_dHpublickey, vc_eMAID));

                 vc_signature := fx_getSignatureFromSA();

                 setverdict(pass,"CertificateUpdateReq message was correct.");
                 vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
                 tc_CMN_ProcessingTime_Timer.stop;
            }
           }
        }
    }
}
[] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_003(
                                vc_SessionID,
                                md_CMN_Signature_002(
                                mw_CMN_SignedInfoType_003)),
                                md_CMN_BodyType_001(
                                mdw_EVCC_CMN_CertificateUpdateReq_001(
                                m_CMN_contractSignatureCertChainWrongOrder_001()))))
                                -> value v_requestMessage {
    tc_V2G_SECC_Sequence_Timer.stop;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            setverdict(v_vct,"Invalid order for contract certificate chain detected " &
                             "(see section 8.5.2.5 in ISO 15118-2). " &
                             "Please change the certificate order according to " &
                             "'Contract leaf certificate' -> 'MO sub2 certificate' " &
                             "-> 'MO sub1 certificate' in the SUT.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                       "was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {certificateUpdateReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received CertificateUpdateReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
        }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_CertificateUpdate_002(urn_iso_15118_2_2013_MsgDataTypes.
                                                 ResponseCodeType responseCode)
                                                 runs on EVCC_Tester return verdicttype  {

    var charstring v_message;

      f_EVCC_setPwmMode(e_PosVolt12);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_EVCC_CMN_CertificateUpdateRes_001(
                                 responseCode, md_CertificateChainType_001(
                                 vc_sAProvisioningCertChain.x509Certificate),
                                 md_CertificateChainType_001(
                                 vc_contractSignatureCertChain.x509Certificate),
                                 vc_contractSignatureEncryptedPrivateKey,
                                 vc_dHpublickey, vc_eMAID))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_PaymentDetailsReq_001(?,?)))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received PaymentDetailsReq message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_CertificateUpdate_003() runs on EVCC_Tester return verdicttype  {

    timer t_wait;

    // Wait until SUT message timeout
    log("Proccessing time: " & toString<float>(vc_processingTime) & "s.");
    t_wait.start(par_V2G_EVCC_Msg_Timeout_CertificateUpdateReq -
                 par_CMN_Transmission_Delay - vc_processingTime);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));

        alt {
            [] t_wait.timeout;
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                          "terminated by the SUT " &
                                                          "before timeout has expired.");
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
                setverdict(fail, "No V2G message was expected.");
            }
        }

        if(getverdict == pass) {


tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_CertificateUpdate_004() runs on EVCC_Tester return verdicttype {

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_CertificateUpdate_005(HAL_61851_PwmMode_Type pwmMode)
                                                 runs on EVCC_Tester
                                                 return verdicttype  {
        var boolean v_tcpTerminated := false;
        timer t_wait;

            f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
            f_EVCC_setPwmMode(pwmMode);
            t_wait.start(par_CMN_waitForNextHAL);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            alt {
               [] t_wait.timeout;
               [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                            "terminated by the SUT " &
                                                            "before request message " &
                                                            "was sent by the Test System. " &
                                                            "The send operation will be " &
                                                            "canceled.") {
                     v_tcpTerminated := true;
               }
            }

        if(not v_tcpTerminated) {
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_003(vc_SessionID,
                                                                     vc_signature),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_CertificateUpdateRes_001(
                                        oK, vc_sAProvisioningCertChain,
                                        vc_contractSignatureCertChain,
                                        vc_contractSignatureEncryptedPrivateKey,
                                        vc_dHpublickey, vc_eMAID))));

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_EVCC_CMN_PaymentDetailsReq_001(
                                            ?,m_CMN_contractSignatureCertChain_001()))) {

                   setverdict(fail,"Received PaymentDetailsReq message was not expected. " &
                                   "CP State E/F should be detected before.");
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }
}
```

I.3.2.8 EVCC functions for CertificateHandlingAndPaymentDetails

```
module TestBehavior_EVCC_CertificateHandlingAndPaymentDetails {

    import from Timer_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Templates_EVCC_CertificateInstallation all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_EVCC_PaymentDetails all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" {type V2G_Message;}
    import from Templates_EVCC_CertificateUpdate all;
    import from Templates_EVCC_CertificateInstallation all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Templates_EVCC_PaymentDetails all;
import from Pics_15118 all;
import from Pixit_15118_2 all;
import from Services_HAL_61851_1 all;
import from Templates_CMN_Authorization all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from Services_Security all;
import from Templates_CMN_Security all;
import from urn_iso_15118_2_2013_X509Def all;
import from http_www_w3_org_2000_09_xmldsig all;
import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
import from LibFunctions_15118_2 all;
import from DataStructure_HAL_61851 all;
import from TTlibrary_Logging all;
import from Timer_15118 all;

function f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_001(in verdicttype v_vct)
                                                                    runs on EVCC_Tester
                                                                    return verdicttype {

    var charstring v_message;
    var V2G_Message v_requestMessage;
    var boolean isSignatureOk := true;
    var boolean v_isAuthentic := false;
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    var boolean isCertInstall := false;
    var boolean isCertUpdate := false;
    var float v_processingDummy := 5.0;

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    alt {
         []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_PaymentDetailsReq_001(
                                          ?,m_CMN_contractSignatureCertChain_001()))))
                                          -> value v_requestMessage {
            tc_V2G_SECC_Sequence_Timer.stop;
            tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
            if(PIXIT_EVCC_CMN_CertificateHandling == certInstall and not isCertInstall) {
                setverdict(inconc,"Please check configuration of the SUT. " &
                                "No certificate installation " &
                                "was triggered by the SUT.");
            } else if(PIXIT_EVCC_CMN_CertificateHandling == certUpdate and not isCertUpdate) {
                setverdict(inconc,"Please check configuration of the SUT. " &
                                "No certificate update " &
                                "was triggered by the SUT.");
            }

            var EMAIDType_1 v_eMAID := v_requestMessage.body.bodyElement.
                                    paymentDetailsReq.eMAID;
            vc_contractSignatureCertChain := v_requestMessage.body.bodyElement.
                                          paymentDetailsReq.
                                          contractSignatureCertChain;

             var boolean result := fx_checkCertificateValidity(vc_contractSignatureCertChain.
                                                               x509Certificate.tBSCertificate.
                                                               validity.not_Before,
                                                               vc_contractSignatureCertChain.
                                                               x509Certificate.tBSCertificate.
                                                               validity.not_After);
            if(not result) {
                    setverdict(v_vct,"Invalid Certificate validity was detected.");
            }
            if(getverdict == pass) {
                log("EMAID: " & toString<charstring>(v_eMAID));
                result := fx_validateEMAID(v_eMAID);
                if(not result) {
                        setverdict(v_vct,"Invalid EMAID syntax was detected.");
                }
                else {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            setverdict(pass,"PaymentDetailsReq message was correct.");
        }
    }
    vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
    tc_CMN_ProcessingTime_Timer.stop;
}
[]pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                            md_CMN_V2G_Message_001(
                           md_CMN_MessageHeaderType_001(
                           vc_SessionID),
                           md_CMN_BodyType_001(
                           md_EVCC_CMN_PaymentDetailsReq_001(
                           ?,m_CMN_contractSignatureCertChainWrongOrder_001()))))
                           -> value v_requestMessage {
    tc_V2G_SECC_Sequence_Timer.stop;
    setverdict(v_vct,"Invalid order for contract certificate chain detected " &
                     "(see section 8.5.2.5 in ISO 15118-2). " &
                     "Please change the certificate order according to " &
                     "'Contract leaf certificate' -> 'MO sub2 certificate' " &
                     "-> 'MO sub1 certificate' in the SUT.");
}
//TestBehavior for CertificateInstallation
[]pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_003(vc_SessionID,
                                   md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                   md_CMN_BodyType_001(
                                   mdw_EVCC_CMN_CertificateInstallationReq_001(
                                   m_CMN_oEMProvisioningCertificate_001()))))
                                   -> value v_requestMessage {
  tc_V2G_SECC_Sequence_Timer.stop;
  isCertInstall := true;
  vc_oEMProvisioningCert := v_requestMessage.body.bodyElement.
                              certificateInstallationReq.oEMProvisioningCert;

  var Signature authSignature := v_requestMessage.header.signature_;

  var BodyType certificateInstallationReqBodyType := v_requestMessage.
                                                     body;

  f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                            cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

  var boolean v_result := fx_validateCertID(vc_oEMProvisioningCert.
                                          tBSCertificate.subject);
  if(not v_result) {
       setverdict(v_vct, "Invalid certID was detected within OEM provisioning " &
                         "certificate.");
  }
  if(getverdict == pass) {
          isSignatureOk := f_verifySignature(
                           certificateInstallationReqBodyType,
                           authSignature,
                           vc_oEMProvisioningCert, v_vct);

      if(isSignatureOk) {
         v_isAuthentic := fx_checkAuthOfOEMProvisioningCert(
                          v_requestMessage);
      } else {
         v_isAuthentic := true;
      }

      if(not v_isAuthentic) {
              setverdict(v_vct,"Authentication of oEMProvisioningCert " &
                               "was not correct.");
       }
       if(v_isAuthentic) {

           f_validateRootCertificateIDLength(certificateInstallationReqBodyType, v_vct);
           if(getverdict == pass) {

               vc_sAProvisioningCertChain :=
               fx_getSAProvisioningCertificateChainFromSA(vc_oEMProvisioningCert);
               vc_contractSignatureCertChain :=
                                     fx_getContractSignatureCertChainFromSA(
                                     vc_oEMProvisioningCert);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                vc_contractSignatureEncryptedPrivateKey :=
                fx_getContractSignatureEncryptedPrivateKeyFromSA(
                vc_oEMProvisioningCert);
                vc_dHpublickey := fx_getDHpublickeyFromSA(vc_oEMProvisioningCert);
                vc_eMAID := fx_getEMAIDFromSA(vc_oEMProvisioningCert);

                var BodyType myCertificateInstallationBodyType := md_CMN_BodyType_001(
                                        md_EVCC_CMN_CertificateInstallationRes_001(
                                        oK, vc_sAProvisioningCertChain,
                                        vc_contractSignatureCertChain,
                                        vc_contractSignatureEncryptedPrivateKey,
                                        vc_dHpublickey, vc_eMAID));


                vc_signature := fx_getSignatureFromSA();

                tc_V2G_SECC_Sequence_Timer.start;
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_003(vc_SessionID,
                                                                         vc_signature),
                                            myCertificateInstallationBodyType));

                setverdict(pass,"CertificateInstallationReq message was correct. " &
                                "Waiting for PaymentDetailsReq message.");
                repeat;
            }
          }
    }
}
[]pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_003(vc_SessionID,
                                md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                md_CMN_BodyType_001(
                                mdw_EVCC_CMN_CertificateUpdateReq_001(
                                m_CMN_contractSignatureCertChain_001()))))
                                -> value v_requestMessage {
    tc_V2G_SECC_Sequence_Timer.stop;
    isCertUpdate := true;
    vc_contractSignatureCert := v_requestMessage.body.bodyElement.certificateUpdateReq.
                                contractSignatureCertChain.x509Certificate;
    vc_contractSignatureCertChain := v_requestMessage.body.bodyElement.
                                        certificateUpdateReq.
                                      contractSignatureCertChain;
    var EMAIDType_1 v_eMAID := v_requestMessage.body.bodyElement.
                              certificateUpdateReq.eMAID;
    var Signature authSignature := v_requestMessage.header.signature_;
    var BodyType certificateUpdateReqBodyType := v_requestMessage.
                                                 body;

    f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                             cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

    if(getverdict == pass) {
            isSignatureOk := f_verifySignature(
                             certificateUpdateReqBodyType,
                             authSignature,
                             vc_contractSignatureCert, v_vct);

        var boolean result := fx_checkCertificateValidity(vc_contractSignatureCertChain.
                                                          x509Certificate.tBSCertificate.
                                                          validity.not_Before,
                                                          vc_contractSignatureCertChain.
                                                          x509Certificate.tBSCertificate.
                                                          validity.not_After);
        if(not result) {
                    setverdict(v_vct,"Invalid Certificate validity was detected.");
        }
        if(getverdict == pass) {
            log("EMAID: " & toString<charstring>(v_eMAID));
            result := fx_validateEMAID(v_eMAID);
            if(not result) {
                      setverdict(v_vct,"Invalid EMAID syntax was detected.");
            }
            if(isSignatureOk and getverdict == pass) {
               v_isAuthentic :=  fx_updateContractSignatureCertChain(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                v_requestMessage);
                } else {
                   v_isAuthentic := true;
                }

                if(not v_isAuthentic) {
                       setverdict(v_vct,"Update of contractSignatureCertChain " &
                                        "was not successful.");
                if(v_isAuthentic) {

                 f_validateRootCertificateIDLength(certificateUpdateReqBodyType, v_vct);
                 if(getverdict == pass) {

                     vc_sAProvisioningCertChain :=
                     fx_getSAProvisioningCertificateChainFromSA(vc_contractSignatureCert);
                     vc_contractSignatureCertChain := fx_getContractSignatureCertChainFromSA(
                                                       vc_contractSignatureCert);
                     vc_contractSignatureEncryptedPrivateKey :=
                     fx_getContractSignatureEncryptedPrivateKeyFromSA(
                     vc_contractSignatureCert);
                     vc_dHpublickey := fx_getDHpublickeyFromSA(vc_contractSignatureCert);

                     vc_eMAID := fx_getEMAIDFromSA(vc_contractSignatureCert);

                     var BodyType myCertificateUpdateBodyType := md_CMN_BodyType_001(
                                                md_EVCC_CMN_CertificateUpdateRes_001(
                                                oK, vc_sAProvisioningCertChain,
                                                vc_contractSignatureCertChain,
                                                vc_contractSignatureEncryptedPrivateKey,
                                                vc_dHpublickey, vc_eMAID));

                     vc_signature := fx_getSignatureFromSA();

                     tc_V2G_SECC_Sequence_Timer.start;
                     pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_003(vc_SessionID,
                                                                           vc_signature),
                                              myCertificateUpdateBodyType));

                     setverdict(pass,"CertificateUpdateReq message was correct. " &
                                    "Waiting for PaymentDetailsReq message.");
                     repeat;
                 }
               }
            }
        }
    }
    []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                    md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_003(
                                    vc_SessionID,
                                    md_CMN_Signature_002(
                                    mw_CMN_SignedInfoType_003)),
                                    md_CMN_BodyType_001(
                                    mdw_EVCC_CMN_CertificateUpdateReq_001(
                                    m_CMN_contractSignatureCertChainWrongOrder_001()))))
                                    -> value v_requestMessage {
        tc_V2G_SECC_Sequence_Timer.stop;
        setverdict(v_vct,"Invalid order for contract certificate chain detected " &
                         "(see section 8.5.2.5 in ISO 15118-2). " &
                         "Please change the certificate order according to " &
                         "'Contract leaf certificate' -> 'MO sub2 certificate' " &
                         "-> 'MO sub1 certificate' in the SUT.");
    }
    [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                   "terminated by the SUT.");
    [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var3(v_vct, "Unexpected V2G message was received.",
                                              {paymentDetailsReq := ?},
                                              {certificateInstallationReq := ?},
                                              {certificateUpdateReq := ?});
    [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                 "but cannot be decoded.");
    [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PaymentDetailsReq or " &
                                              "CertificateInstallationReq or " &
                                              "CertificateUpdateReq " &
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                        "message, but with invalid " &
                                                        "data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_002(ResponseCodeType
                                                                    responseCode)
                                                                    runs on EVCC_Tester
                                                                    return verdicttype {

    var charstring v_message;

      f_EVCC_setPwmMode(e_PosVolt12);
    vc_GenChallenge := fx_generateGenChallenge();
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_EVCC_CMN_PaymentDetailsRes_001(
                                 responseCode,vc_GenChallenge,
                                 fx_getTimeAsLong()))));

    tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_AuthorizationRes_001(?,?)))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received AuthorizationRes message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_003() runs on EVCC_Tester
                                                                      return verdicttype {

    timer t_wait;

    // Wait until SUT message timeout
    log("Proccessing time: " & toString<float>(vc_processingTime) & "s.");
    t_wait.start(par_V2G_EVCC_Msg_Timeout_PaymentDetailsReq -
                 par_CMN_Transmission_Delay - vc_processingTime);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_004() runs on EVCC_Tester
                                                                          return verdicttype {

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_CertificateHandlingAndPaymentDetails_005(
                                            HAL_61851_PwmMode_Type pwmMode)
                                            runs on EVCC_Tester
                                            return verdicttype {
        var boolean v_tcpTerminated := false;
        timer t_wait;

            f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
            f_EVCC_setPwmMode(pwmMode);
            t_wait.start(par_CMN_waitForNextHAL);
            alt {
               [] t_wait.timeout;
               [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                             "terminated by the SUT " &
                                                             "before request message " &
                                                             "was sent by the Test System. " &
                                                             "The send operation will be " &
                                                             "canceled.") {
                   v_tcpTerminated := true;
               }
            }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(not v_tcpTerminated) {
            vc_GenChallenge:=fx_generateGenChallenge();
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_PaymentDetailsRes_001(
                                        oK,vc_GenChallenge,
                                        fx_getTimeAsLong()))));

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                             "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(?),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_CMN_AuthorizationReq_002(
                                                 ?,vc_GenChallenge)))) {

                   setverdict(fail,"Received AuthorizationReq message was not expected. " &
                                   "CP State E/F should be detected before.");
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }
}
```

I.3.2.9 EVCC functions for PaymentDetails

```
module TestBehavior_EVCC_PaymentDetails {

    import from Timer_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts {type EVCC_Tester;};
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_EVCC_PaymentDetails all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" {type V2G_Message;}
    import from Templates_EVCC_CertificateUpdate all;
    import from Templates_EVCC_CertificateInstallation all;
    import from Services_HAL_61851_1 all;
    import from Services_Security all;
    import from Templates_CMN_Security all;
    import from LibFunctions_15118_2 all;
    import from TTlibrary_Logging all;

    function f_EVCC_CMN_TB_VTB_PaymentDetails_001() runs on EVCC_Tester return verdicttype  {

        var charstring v_message;
        var V2G_Message v_requestMessage;
        tc_V2G_SECC_Sequence_Timer.start;
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_003(vc_SessionID,
                                                                 vc_signature),
                                    md_CMN_BodyType_001(
```

 © ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                md_EVCC_CMN_CertificateInstallationRes_001(
                                oK, vc_sAProvisioningCertChain,
                                vc_contractSignatureCertChain,
                                vc_contractSignatureEncryptedPrivateKey,
                                vc_dHpublickey, vc_eMAID))));

    alt {
        []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_PaymentDetailsReq_001(
                                        ?,m_CMN_contractSignatureCertChain_001()))))
                                        -> value v_requestMessage {
            tc_V2G_SECC_Sequence_Timer.stop;
            var EMAIDType_1 v_eMAID := v_requestMessage.body.bodyElement.paymentDetailsReq.eMAID;
            vc_contractSignatureCertChain := v_requestMessage.body.bodyElement.paymentDetailsReq.
                                            contractSignatureCertChain;

            var boolean result := fx_checkCertificateValidity(vc_contractSignatureCertChain.
                                                              x509Certificate.tBSCertificate.
                                                              validity.not_Before,
                                                              vc_contractSignatureCertChain.
                                                              x509Certificate.tBSCertificate.
                                                              validity.not_After);
            if(not result) {
                setverdict(fail,"Invalid Certificate validity was detected.");
            }
            if(getverdict == pass) {
                log("EMAID: " & toString<charstring>(v_eMAID));
                result := fx_validateEMAID(v_eMAID);
                if(not result) {
                    setverdict(fail,"Invalid EMAID syntax was detected.");
                }
                else {
                    setverdict(pass,"PaymentDetailsReq message was correct.");
                }
            }
        }
        []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentDetailsReq_001(
                                    ?,m_CMN_contractSignatureCertChainWrongOrder_001()))))
                                    -> value v_requestMessage {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Invalid order for contract certificate chain detected " &
                            "(see section 8.5.2.5 in ISO 15118-2). " &
                            "Please change the certificate order according to " &
                            "'Contract leaf certificate' -> 'MO sub2 certificate' " &
                            "-> 'MO sub1 certificate' in the SUT.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                    {paymentDetailsReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentDetailsReq message, " &
                                                   "but with invalid data content.");
        [] a_EVCC_Timeout(fail);
        }
     return getverdict;
  }

function f_EVCC_CMN_TB_VTB_PaymentDetails_002() runs on EVCC_Tester return verdicttype  {

    var charstring v_message;
    var V2G_Message v_requestMessage;
    tc_V2G_SECC_Sequence_Timer.start;
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_003(vc_SessionID,
                                                             vc_signature),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_CertificateUpdateRes_001(
                                oK, vc_sAProvisioningCertChain,
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                vc_contractSignatureCertChain,
                                vc_contractSignatureEncryptedPrivateKey,
                                vc_dHpublickey, vc_eMAID))));

        alt {
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_PaymentDetailsReq_001(
                                        ?,m_CMN_contractSignatureCertChain_001()))))
                                        -> value v_requestMessage {
                tc_V2G_SECC_Sequence_Timer.stop;

                var EMAIDType_1 v_eMAID := v_requestMessage.body.bodyElement.paymentDetailsReq.eMAID;
                vc_contractSignatureCertChain := v_requestMessage.body.bodyElement.paymentDetailsReq.
                                                 contractSignatureCertChain;

                var boolean result := fx_checkCertificateValidity(vc_contractSignatureCertChain.
                                                                  x509Certificate.tBSCertificate.
                                                                  validity.not_Before,
                                                                  vc_contractSignatureCertChain.
                                                                  x509Certificate.tBSCertificate.
                                                                  validity.not_After);
                if(not result) {
                    setverdict(fail,"Invalid Certificate validity was detected.");
                }
                if(getverdict == pass) {
                    log("EMAID: " & toString<charstring>(v_eMAID));
                    result := fx_validateEMAID(v_eMAID);
                    if(not result) {
                        setverdict(fail,"Invalid EMAID syntax was detected.");
                    }
                    else {
                        setverdict(pass,"PaymentDetailsReq message was correct.");
                    }
                }
            }
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_PaymentDetailsReq_001(
                                        ?,m_CMN_contractSignatureCertChainWrongOrder_001()))))
                                        -> value v_requestMessage {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(fail,"Invalid order for contract certificate chain detected " &
                                "(see section 8.5.2.5 in ISO 15118-2). " &
                                "Please change the certificate order according to " &
                                "'Contract leaf certificate' -> 'MO sub2 certificate' " &
                                "-> 'MO sub1 certificate' in the SUT.");
            }
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                          "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {paymentDetailsReq := ?});
            [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentDetailsReq message, " &
                                                       "but with invalid data content.");
            [] a_EVCC_Timeout(fail);
            }
         return getverdict;
      }

    function f_EVCC_CMN_TB_VTB_PaymentDetails_003() runs on EVCC_Tester return verdicttype  {

        var charstring v_message;
        var V2G_Message v_requestMessage;
        tc_V2G_SECC_Sequence_Timer.start;
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

        alt {
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_PaymentDetailsReq_001(
                                        ?,m_CMN_contractSignatureCertChain_001()))))
                                        -> value v_requestMessage {
            tc_V2G_SECC_Sequence_Timer.stop;

            var EMAIDType_1 v_eMAID := v_requestMessage.body.bodyElement.paymentDetailsReq.eMAID;
            vc_contractSignatureCertChain := v_requestMessage.body.bodyElement.paymentDetailsReq.
                                              contractSignatureCertChain;

            var boolean result := fx_checkCertificateValidity(vc_contractSignatureCertChain.
                                                              x509Certificate.tBSCertificate.
                                                              validity.not_Before,
                                                              vc_contractSignatureCertChain.
                                                              x509Certificate.tBSCertificate.
                                                              validity.not_After);
            if(not result) {
                setverdict(fail,"Invalid Certificate validity was detected.");
            }
            if(getverdict == pass) {
                log("EMAID: " & toString<charstring>(v_eMAID));
                result := fx_validateEMAID(v_eMAID);
                if(not result) {
                    setverdict(fail,"Invalid EMAID syntax was detected.");
                }
                else {
                    setverdict(pass,"PaymentDetailsReq message was correct.");
                }
            }
        }
        []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentDetailsReq_001(
                                ?,m_CMN_contractSignatureCertChainWrongOrder_001()))))
                                -> value v_requestMessage {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Invalid order for contract certificate chain detected " &
                            "(see section 8.5.2.5 in ISO 15118-2). " &
                            "Please change the certificate order according to " &
                            "'Contract leaf certificate' -> 'MO sub2 certificate' " &
                            "-> 'MO sub1 certificate' in the SUT.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                {paymentDetailsReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PaymentDetailsReq message, " &
                                                   "but with invalid data content.");
        [] a_EVCC_Timeout(fail);
        }
    return getverdict;
  }
}
```

I.3.2.10 EVCC functions for Authorization

```
module TestBehavior_EVCC_Authorization {

    import from Pixit_15118 all;
    import from Templates_CMN_Security all;
    import from Timer_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Services_HAL_61851_1 all;
    import from ComponentsAndPorts all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Templates_CMN_Authorization all;
import from Templates_EVCC_PaymentDetails all;
import from Templates_EVCC_PaymentServiceSelection all;
import from Templates_EVCC_ChargeParameterDiscovery all;
import from Templates_CMN_V2GMessage all;
import from Pics_15118_2 all;
import from Pics_15118 all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from http_www_w3_org_2000_09_xmldsig all;
import from Services_Security all;
import from LibFunctions_15118_2 all;
import from DataStructure_HAL_61851 all;
import from Timer_15118 all;

function f_EVCC_CMN_TB_VTB_Authorization_001(in verdicttype v_vct)
                                             runs on EVCC_Tester
                                             return verdicttype {

    var charstring v_message;
    var V2G_Message v_requestMessage;
    var boolean isSignatureOk := true;

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    if(PICS_CMN_CMN_IdentificationMode == pnC){
        vc_GenChallenge:=fx_generateGenChallenge();
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(md_EVCC_CMN_PaymentDetailsRes_001(
                                    oK,vc_GenChallenge,
                                    fx_getTimeAsLong())))));
    }else {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));
    }
    alt{
        [PICS_CMN_CMN_IdentificationMode == pnC]
            pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_003(vc_SessionID,
                                md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                md_CMN_BodyType_001(md_CMN_CMN_AuthorizationReq_002(
                                ?,vc_GenChallenge)))) -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            var Signature authSignature := v_requestMessage.header.signature_;
            var BodyType authorizationReqBodyType := v_requestMessage.body;

            f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                        cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

            if(getverdict == pass) {


                isSignatureOk := f_verifySignature(authorizationReqBodyType,
                                                   authSignature,
                                                   vc_contractSignatureCertChain.
                                                   x509Certificate, v_vct);
                if(isSignatureOk) {
                    setverdict(pass, "AuthorizationReq message was correct.");
                }
                else {
                    setverdict(v_vct, "Invalid signature of the AuthorizationReq " &
                                      "message was received.");
                }
            }
        }
        [PICS_CMN_CMN_IdentificationMode == eIM]
          pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         m_CMN_CMN_AuthorizationReq_001()))) {

            tc_V2G_SECC_Sequence_Timer.stop;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            setverdict(pass, "AuthorizationReq message was correct.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                       "misleadingly terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {authorizationReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received AuthorizationReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_Authorization_002(in verdicttype v_vct)
                                             runs on EVCC_Tester
                                             return verdicttype {

    var charstring v_message;
    timer t_ongoing;
    var integer v_cnt := 0;
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    if(PICS_CMN_CMN_IdentificationMode == eIM){
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));
    }else {
        setverdict(inconc, "Authorization: Check PICS_CMN_CMN_IdentificationMode");
    }
    alt {
        [] t_ongoing.timeout {
            alt {
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_002(
                                                vc_SessionID),
                                                md_CMN_BodyType_001(
                                                m_CMN_CMN_AuthorizationReq_001()))) {

                    tc_V2G_SECC_Sequence_Timer.stop;
                    setverdict(pass, "AuthorizationReq message was correct.");
                }
                [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                               "misleadingly terminated " &
                                                               "by the SUT.");
                [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {authorizationReq := ?});
                [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                             "but cannot be decoded.");
                [] a_EVCC_Unexpected_Message_Content(v_vct, "Received AuthorizationReq " &
                                                            "message, but with invalid " &
                                                            "data content.");
                [] a_EVCC_Timeout(v_vct);
            }
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        m_CMN_CMN_AuthorizationReq_001()))) {

            tc_V2G_SECC_Sequence_Timer.stop;
            if(v_cnt == 0) {
                t_ongoing.start(par_SECC_timeAuthenticationLoop);
            }
            v_cnt := v_cnt + 1;
            setverdict(pass, "AuthorizationReq message was correct.");

            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_AuthorizationRes_001(oK,
                                    ongoing_WaitingForCustomerInteraction))));
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            repeat;
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                       "misleadingly terminated " &
                                                       "by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {authorizationReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received AuthorizationReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }

    return getverdict;
}

function f_EVCC_CMN_TB_VTB_Authorization_003(ResponseCodeType responseCode)
                                          runs on EVCC_Tester return verdicttype  {

    var charstring v_message;
      f_EVCC_setPwmMode(e_PosVolt12);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(md_CMN_CMN_AuthorizationRes_001(
                                 responseCode, finished))));

    tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                         *,?,?)))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received ChargeParamterDiscoveryReq message " &
                            "was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_Authorization_004() runs on EVCC_Tester return verdicttype  {

    timer t_wait;

    // Wait until SUT message timeout
    t_wait.start(par_V2G_EVCC_Msg_Timeout_AuthorizationReq -
                 par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }

        if(getverdict == pass) {

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_Authorization_005() runs on EVCC_Tester return verdicttype  {

        var V2G_Message v_requestMessage;
        var boolean isSignatureOk := true;

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);


        if(PICS_CMN_CMN_IdentificationMode == pnC){
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_AuthorizationRes_001(oK, ongoing))));
        }
        else {
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(md_CMN_CMN_AuthorizationRes_001(
                                        oK, ongoing_WaitingForCustomerInteraction))));
        }
        tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout - par_CMN_Transmission_Delay);
        alt
        {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                           md_CMN_MessageHeaderType_001(vc_SessionID),
                                           md_CMN_BodyType_001(
                                           m_CMN_CMN_AuthorizationReq_001))) {

                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass, "AuthorizationReq message was correct.");
                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                if(PICS_CMN_CMN_IdentificationMode == pnC){
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(
                                                 vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_CMN_AuthorizationRes_001(
                                                 oK, ongoing))));
                }
                else {
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_CMN_AuthorizationRes_001(
                                                 oK, ongoing_WaitingForCustomerInteraction))));
                }
                repeat;
            }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                        "terminated by the SUT before " &
                                        "timeout has expired.");
        [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                       {authorizationReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                            "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received AuthorizationReq message, " &
                                           "but with invalid data content.");
        [] a_EVCC_Timeout(fail);
        []tc_V2G_EVCC_Ongoing_Timer.timeout{

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                        "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        m_CMN_CMN_AuthorizationReq_001)))
                {
                    tc_V2G_SECC_Sequence_Timer.stop;
                    setverdict(pass, "AuthorizationReq message was correct.");
                    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                    if(PICS_CMN_CMN_IdentificationMode == pnC){
                        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(
                                                vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_AuthorizationRes_001(
                                                oK, ongoing))));
                    }
                    else {
                        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_AuthorizationRes_001(
                                                oK, ongoing_WaitingForCustomerInteraction))));
                    }
                    repeat;
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                     "but TCP connection " &
                                                     "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                 "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_Authorization_006() runs on EVCC_Tester return verdicttype {

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_Authorization_007(HAL_61851_PwmMode_Type pwmMode)
                                             runs on EVCC_Tester
                                             return verdicttype  {
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_AuthorizationRes_001(oK, finished))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                         *,?,?)))) {
               setverdict(fail,"Received ChargeParamterDiscoveryReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_Authorization_008(in verdicttype v_vct)
                                             runs on EVCC_Tester
                                             return verdicttype {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var charstring v_message;
var V2G_Message v_requestMessage;
var boolean isSignatureOk := true;
timer t_ongoing;
var integer v_cnt := 0;

tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
if(PICS_CMN_CMN_IdentificationMode == pnC){
    vc_GenChallenge:=fx_generateGenChallenge();
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(md_EVCC_CMN_PaymentDetailsRes_001(
                                oK,vc_GenChallenge,
                                fx_getTimeAsLong())))));
}else {
    setverdict(inconc, "Authorization: Check PICS_CMN_CMN_IdentificationMode");
}
alt{
    [] t_ongoing.timeout {
        alt {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_002(
                                                 vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 m_CMN_CMN_AuthorizationReq_001())))
                                                 -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass, "AuthorizationReq message was correct.");
            }
            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                        "misleadingly terminated " &
                                                        "by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                         {authorizationReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                   "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received AuthorizationReq " &
                                                  "message, but with invalid " &
                                                  "data content.");
            [] a_EVCC_Timeout(v_vct);
        }
    }
    [v_cnt == 0] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_003(vc_SessionID,
                                      md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                      md_CMN_BodyType_001(md_CMN_CMN_AuthorizationReq_002(
                                      ?,vc_GenChallenge)))) -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        v_cnt := v_cnt + 1;
        var Signature authSignature := v_requestMessage.header.signature_;
        var BodyType authorizationReqBodyType := v_requestMessage.body;

        t_ongoing.start(par_SECC_timeAuthenticationLoop);

        f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                    cc_algorithm_method_2,
                                    cc_algorithm_method_3, v_vct);

        if(getverdict == pass) {


                isSignatureOk := f_verifySignature(authorizationReqBodyType,
                                                   authSignature,
                                                   vc_contractSignatureCertChain.
                                                   x509Certificate, v_vct);
            if(isSignatureOk) {
                setverdict(pass, "AuthorizationReq message was correct.");
            }
            else {
                setverdict(v_vct, "Invalid signature of the AuthorizationReq " &
                                  "message was received.");
            }
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_AuthorizationRes_001(oK, ongoing))));
                repeat;
            }
            [v_cnt > 0] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_002(vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  m_CMN_CMN_AuthorizationReq_001())))
                                                  -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                v_cnt := v_cnt + 1;
                setverdict(pass, "AuthorizationReq message was correct.");

                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_AuthorizationRes_001(oK, ongoing))));
                repeat;
            }

            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                           "misleadingly terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {authorizationReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received AuthorizationReq message, " &
                                                        "but with invalid data content.");
            [] a_EVCC_Timeout(v_vct);
        }

        return getverdict;
    }
}
```

I.3.2.11 EVCC functions for ChargeParameterDiscovery

```
module TestBehavior_EVCC_ChargeParameterDiscovery {

    import from Timer_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from ComponentsAndPorts all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Templates_EVCC_ChargeParameterDiscovery all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_CMN_Authorization all;
    import from Templates_CMN_CableCheck all;
    import from Templates_CMN_V2GMessage all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from Pics_15118 all;
    import from Pixit_15118_2 all;
    import from Pixit_15118 all;
    import from Services_HAL_61851_1 all;
    import from TTlibrary_Logging all;
    import from Templates_CMN_PreCharge all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from Timer_15118 all;
    import from DataStructure_V2G all;
    import from Pics_15118_2 all;
    import from Services_Security all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from XSDAUX all;
import from TestBehavior_EVCC_PowerDelivery all;

function f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_001(in verdicttype v_vct)
                                                        runs on EVCC_Tester
                                                        return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;
    var float v_processingDummy := 5.0;

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
      pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_CMN_AuthorizationRes_001(oK, finished))));

     alt {
          []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                          *,?,?)))) -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
                if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                             maxEntriesSAScheduleTuple)) {
                    vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                                    chargeParameterDiscoveryReq.
                                                    maxEntriesSAScheduleTuple;
                }
                vc_requestedEnergyTransferMode := v_requestMessage.body.bodyElement.
                                                  chargeParameterDiscoveryReq.
                                                  requestedEnergyTransferMode;
                var EVChargeParameter_group evChargeParameter := v_requestMessage.
                                                                 body.bodyElement.
                                                                 chargeParameterDiscoveryReq.
                                                                 eVChargeParameter;
                if(ischosen(evChargeParameter.aC_EVChargeParameter)){
                    vc_EAmount := evChargeParameter.aC_EVChargeParameter.eAmount;
                    f_check_PhysicalValue_valueRange(cc_eAmount, vc_EAmount, wh,
                                                     cc_maximum_value_physicalValue_case4,
                                                     v_vct);
                    vc_MeterInfo.meterReading := float2int(f_CMN_PhysicalValue_GetValue(
                                                           vc_EAmount));
                    vc_EVMaxVoltage := evChargeParameter.aC_EVChargeParameter.eVMaxVoltage;
                    f_check_PhysicalValue_valueRange(cc_eVMaxVoltage, vc_EVMaxVoltage, v,
                                                     cc_maximum_value_physicalValue_case2,
                                                     v_vct);
                    vc_EVMaxCurrent := evChargeParameter.aC_EVChargeParameter.eVMaxCurrent;
                    f_check_PhysicalValue_valueRange(cc_eVMaxCurrent, vc_EVMaxCurrent, a,
                                                     cc_maximum_value_physicalValue_case1,
                                                     v_vct);
                    vc_EVMinCurrent := evChargeParameter.aC_EVChargeParameter.eVMinCurrent;
                    f_check_PhysicalValue_valueRange(cc_eVMinCurrent, vc_EVMinCurrent, a,
                                                     cc_maximum_value_physicalValue_case1,
                                                     v_vct);
                    if((not ispresent(evChargeParameter.aC_EVChargeParameter.departureTime)) and
                        PIXIT_EVCC_CMN_DepartureTime == departureTime) {
                        setverdict(inconc, "A departure time was not selected by the SUT");
                    } else {
                        if(ispresent(evChargeParameter.aC_EVChargeParameter.departureTime)) {
                            vc_DepartureTime := evChargeParameter.aC_EVChargeParameter.
                                                departureTime;
                            if(vc_DepartureTime == 0) {
                               logWarn("A departure time equals to 0s " &
                                           "might lead to interoperability " &
                                       "issues if the EVSE proceeds as " &
                                       "defined in [V2G2-303].");
                            }
                            tc_EVCC_DepartureTime_Timer.start(int2float(vc_DepartureTime));
                        }
                    }
                    if(PICS_CMN_CMN_ChargingMode != aC) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            setverdict(inconc, "Invalid charging mode is selected " &
                               "(DC was expected). " &
                               "Please check PIXIT configuration.");
        }
    }
    else{
        if(evChargeParameter.dC_EVChargeParameter.dC_EVStatus.eVReady == true) {
            logWarn("The field 'evReady' should be set to 'false'.");
        }
        vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                        chargeParameterDiscoveryReq.eVChargeParameter.
                        dC_EVChargeParameter.eVMaximumCurrentLimit;
        f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                         vc_EVMaximumCurrentLimit, a,
                                         cc_maximum_value_physicalValue_case1,
                                         v_vct);
        vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
        chargeParameterDiscoveryReq.eVChargeParameter.dC_EVChargeParameter.
        eVMaximumVoltageLimit;
        f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                         vc_EVMaximumVoltageLimit, v,
                                         cc_maximum_value_physicalValue_case2,
                                         v_vct);
        if(ispresent(v_requestMessage.body.bodyElement.
                     chargeParameterDiscoveryReq.eVChargeParameter.
                     dC_EVChargeParameter.eVMaximumPowerLimit)) {
              vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                        chargeParameterDiscoveryReq.
                                        eVChargeParameter.
                                        dC_EVChargeParameter.eVMaximumPowerLimit;
              f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                               vc_EVMaximumPowerLimit, w,
                                               cc_maximum_value_physicalValue_case4,
                                             v_vct);
              vc_isPowLimit := true;
        }
        log("------------------------------------------------------------");
        log("EV maximum charging limits:");
        log("EVMaximumCurrentLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumCurrentLimit)));
        log("EVMaximumVoltageLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumVoltageLimit)));
        if(vc_isPowLimit) {
            log("EVMaximumPowerLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumPowerLimit)));
        }
        log("------------------------------------------------------------");
        if((not ispresent(evChargeParameter.dC_EVChargeParameter.departureTime)) and
            PIXIT_EVCC_CMN_DepartureTime == departureTime) {
            setverdict(inconc, "A departure time was not selected by the SUT");
        } else {
            if(ispresent(evChargeParameter.dC_EVChargeParameter.departureTime)) {
                vc_DepartureTime := evChargeParameter.dC_EVChargeParameter.
                                    departureTime;
                if(vc_DepartureTime == 0) {
                   logWarn("A departure time equals to 0s " &
                               "might lead to interoperability " &
                           "issues if the EVSE proceeds as " &
                           "defined in [V2G2-303].");
                }
                tc_EVCC_DepartureTime_Timer.start(int2float(vc_DepartureTime));
            }
        }
        if(PICS_CMN_CMN_ChargingMode != dC) {
            setverdict(inconc, "Invalid charging mode is selected " &
                               "(AC was expected). " &
                               "Please check PIXIT configuration.");
        }
    }
    if(getverdict == pass) {
        setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var SASchedules_group v_sASchedules := f_getSchedules(
                                        vc_SAScheduleTupleId,
                                        vc_SAScheduleTupleId2,
                                        vc_SAScheduleTupleId3,
                                        vc_maxEntriesSAScheduleTuple);
vc_SASchedule_List := v_sASchedules.sAScheduleList;

if(PICS_CMN_CMN_IdentificationMode == pnC and vc_useSalesTariffs) {

    if(PICS_CMN_CMN_ChargingMode == aC) {

        vc_myChargeParameterBodyType := md_CMN_BodyType_001(
                              md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                              oK, finished, v_sASchedules,
                              m_EVCC_AC_EVSEChargeParameter_001(
                              vc_EVSENominalVoltage,
                              vc_EVSEMaxCurrent)));

        vc_salesTariffIdDigestValueMap :=
                  fx_getIdDigestMapFromScheduleListType(
                  vc_SASchedule_List,
                  par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
                  - cc_salesTariff_processingTime_remains);

        if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                              md_CMN_MessageHeaderType_001(vc_SessionID),
                              md_CMN_BodyType_001(
                              md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                              oK, ongoing, omit,
                              m_EVCC_AC_EVSEChargeParameter_001(
                              vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));
            repeat;
        }
    } else if(PICS_CMN_CMN_ChargingMode == dC) {
        log("------------------------------------------------------------");
        log("EVSE maximum charging limits:");
        log("EVSEMaximumCurrentLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumCurrentLimit)));
        log("EVSEMaximumVoltageLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumVoltageLimit)));
        log("EVSEMaximumPowerLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumPowerLimit)));
        log("EVSEMinimumCurrentLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMinimumCurrentLimit)));
        log("EVSEMinimumVoltageLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMinimumVoltageLimit)));
        log("------------------------------------------------------------");

        vc_myChargeParameterBodyType := md_CMN_BodyType_001(
                            md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                            oK, finished, v_sASchedules,
                            md_EVCC_DC_EVSEChargeParameter_002(
                            vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                            vc_EVSEMaximumPowerLimit,
                            vc_EVSEMaximumVoltageLimit,
                            vc_EVSEMinimumCurrentLimit,
                            vc_EVSEMinimumVoltageLimit,
                            vc_EVSECurrentRegulationTolerance,
                            vc_EVSEPeakCurrentRipple,
                            vc_EVSEEnergyToBeDelivered)));

        vc_salesTariffIdDigestValueMap :=
                  fx_getIdDigestMapFromScheduleListType(
                  vc_SASchedule_List,
                  par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
                  - cc_salesTariff_processingTime_remains);

        if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                                      md_CMN_BodyType_001(
                                                      md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                                      oK, ongoing, omit,
                                                      md_EVCC_DC_EVSEChargeParameter_002(
                                                      vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                                      vc_EVSEMaximumPowerLimit,
                                                      vc_EVSEMaximumVoltageLimit,
                                                      vc_EVSEMinimumCurrentLimit,
                                                      vc_EVSEMinimumVoltageLimit,
                                                      omit,
                                                      vc_EVSEPeakCurrentRipple,
                                                      omit)))));
                                    repeat;
                                 }
                              }
                           }
                           vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
                           tc_CMN_ProcessingTime_Timer.stop;
                        }
                     }
                     [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                                    "was misleadingly " &
                                                                     "terminated by the SUT.");
                     [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                                     {chargeParameterDiscoveryReq := ?});
                     [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                                  "but cannot be decoded.");
                     [] a_EVCC_Unexpected_Message_Content(v_vct, "Received ChargeParamterDiscoveryReq " &
                                                                  "message, but with invalid " &
                                                                  "data content.");
                     [] a_EVCC_Timeout(v_vct);
                  }
                  return getverdict;
               }

               function f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_002() runs on EVCC_Tester
                                                                    return verdicttype  {

                  timer t_wait;

                  // Wait until SUT message timeout
                  log("Processing time: " & toString<float>(vc_processingTime) & "s.");
                  t_wait.start(par_V2G_EVCC_Msg_Timeout_ChargeParameterDiscoveryReq -
                               par_CMN_Transmission_Delay - vc_processingTime);
                  pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                                  e_getPortStatus));

                  alt {
                     [] t_wait.timeout;
                     [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                                   "terminated by the SUT " &
                                                                   "before timeout has expired.");
                     [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
                          setverdict(fail, "No V2G message was expected.");
                     }
                  }

                  if(getverdict == pass) {

                     tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

                     alt {
                        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                                      "terminated by the SUT.");
                        [] a_EVCC_CMN_SessionStop_Termination() {
                         tc_CMN_TCP_Connection_Termination_Timer.start(
                         par_CMN_TCP_Connection_Termination_Timeout);
                         repeat;
                        }
                        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                                   "but TCP connection " &
                                                                   "was >>NOT<< terminated by the SUT");
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                              "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}


function f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_003() runs on EVCC_Tester return verdicttype {

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_004() runs on EVCC_Tester
                                                          return verdicttype  {

    var V2G_Message v_requestMessage;
    var charstring v_message;
    var integer v_departureTimePrev;
    var integer v_timeElapsed;

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_CMN_CMN_AuthorizationRes_001(oK, finished))));

     alt {
          []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                          *,vc_requestedEnergyTransferMode,?))))
                                          -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                             maxEntriesSAScheduleTuple)) {
                    vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                                    chargeParameterDiscoveryReq.
                                                    maxEntriesSAScheduleTuple;
                }
                var EVChargeParameter_group evChargeParameter := v_requestMessage.
                                                                 body.bodyElement.
                                                                 chargeParameterDiscoveryReq.
                                                                 eVChargeParameter;
                if(ischosen(evChargeParameter.aC_EVChargeParameter)){
                    vc_EAmount := evChargeParameter.aC_EVChargeParameter.eAmount;
                    f_check_PhysicalValue_valueRange(cc_eAmount, vc_EAmount, wh,
                                                     cc_maximum_value_physicalValue_case4,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    fail);
        vc_MeterInfo.meterReading := float2int(f_CMN_PhysicalValue_GetValue(
                                          vc_EAmount));
        vc_EVMaxVoltage := evChargeParameter.aC_EVChargeParameter.eVMaxVoltage;
        f_check_PhysicalValue_valueRange(cc_eVMaxVoltage, vc_EVMaxVoltage, v,
                                    cc_maximum_value_physicalValue_case2,
                                    fail);
        vc_EVMaxCurrent := evChargeParameter.aC_EVChargeParameter.eVMaxCurrent;
        f_check_PhysicalValue_valueRange(cc_eVMaxCurrent, vc_EVMaxCurrent, a,
                                    cc_maximum_value_physicalValue_case1,
                                    fail);
        vc_EVMinCurrent := evChargeParameter.aC_EVChargeParameter.eVMinCurrent;
        f_check_PhysicalValue_valueRange(cc_eVMinCurrent, vc_EVMinCurrent, a,
                                    cc_maximum_value_physicalValue_case1,
                                    fail);
        if(PIXIT_EVCC_CMN_DepartureTime == departureTime) {
            v_departureTimePrev := vc_DepartureTime;
            vc_DepartureTime := evChargeParameter.aC_EVChargeParameter.departureTime;
            v_timeElapsed := float2int(tc_EVCC_DepartureTime_Timer.read);
            tc_EVCC_DepartureTime_Timer.stop;
            if(vc_DepartureTime >= ((v_departureTimePrev - v_timeElapsed) -
                                par_EVCC_DepartureTimeDev) and
               vc_DepartureTime <= ((v_departureTimePrev - v_timeElapsed) +
                                par_EVCC_DepartureTimeDev)) {
               setverdict(pass,"The departure time was reduced by the elapsed time " &
                              "after paused session.");
            }
        }
    }
    else{
        if(evChargeParameter.dC_EVChargeParameter.dC_EVStatus.eVReady == true) {
            logWarn("The field 'evReady' should be set to 'false'.");
        }
        vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                                chargeParameterDiscoveryReq.
                                eVChargeParameter.dC_EVChargeParameter.
                                eVMaximumCurrentLimit;
        f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                     vc_EVMaximumCurrentLimit, a,
                                     cc_maximum_value_physicalValue_case1,
                                     fail);
        vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                                chargeParameterDiscoveryReq.eVChargeParameter.
                                dC_EVChargeParameter.eVMaximumVoltageLimit;
        f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                     vc_EVMaximumVoltageLimit, v,
                                     cc_maximum_value_physicalValue_case2,
                                     fail);
        if(ispresent(v_requestMessage.body.bodyElement.
                     chargeParameterDiscoveryReq.eVChargeParameter.
                     dC_EVChargeParameter.eVMaximumPowerLimit)) {
              vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                    chargeParameterDiscoveryReq.
                                    eVChargeParameter.
                                    dC_EVChargeParameter.
                                    eVMaximumPowerLimit;
              f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                         vc_EVMaximumPowerLimit, w,
                                          cc_maximum_value_physicalValue_case4,
                                          fail);
              vc_isPowLimit := true;
        }
        log("------------------------------------------------------------");
        log("EV maximum charging limits:");
        log("EVMaximumCurrentLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumCurrentLimit)));
        log("EVMaximumVoltageLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumVoltageLimit)));
        if(vc_isPowLimit) {
            log("EVMaximumPowerLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumPowerLimit)));
        }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    log("-------------------------------------------------------------");
                    if(PIXIT_EVCC_CMN_DepartureTime == departureTime) {
                        v_departureTimePrev := vc_DepartureTime;
                        vc_DepartureTime := evChargeParameter.dC_EVChargeParameter.departureTime;
                        v_timeElapsed := float2int(tc_EVCC_DepartureTime_Timer.read);
                        tc_EVCC_DepartureTime_Timer.stop;
                        if(vc_DepartureTime >= ((v_departureTimePrev - v_timeElapsed) -
                                                par_EVCC_DepartureTimeDev) and
                           vc_DepartureTime <= ((v_departureTimePrev - v_timeElapsed) +
                                                par_EVCC_DepartureTimeDev)) {
                           setverdict(pass,"The departure time was reduced by the elapsed time " &
                                           "after paused session.");
                        }
                    }
                }
                if(getverdict == pass) {
                    setverdict(pass,"ChargeParamterDiscoveryReq message was correct. The " &
                                    "previous RequestedEnergyTransferMode was selected after " &
                                    "paused session.");
                }
            }
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                          "was misleadingly " &
                                                         "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {chargeParameterDiscoveryReq := ?});
            [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received ChargeParamterDiscoveryReq " &
                                                       "message, but with invalid " &
                                                       "data content.");
            [] a_EVCC_Timeout(fail);
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_ChargeParameterDiscovery_005() runs on EVCC_Tester
                                                              return verdicttype  {

        var V2G_Message v_requestMessage;
        var charstring v_message;

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_AuthorizationRes_001(oK, finished))));

         alt {
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                          *,vc_requestedEnergyTransferMode,?))))
                                          -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                             maxEntriesSAScheduleTuple)) {
                    vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                                    chargeParameterDiscoveryReq.
                                                    maxEntriesSAScheduleTuple;
                }
                var EVChargeParameter_group evChargeParameter := v_requestMessage.
                                                                 body.bodyElement.
                                                                 chargeParameterDiscoveryReq.
                                                                 eVChargeParameter;
                if(ischosen(evChargeParameter.aC_EVChargeParameter)){
                    vc_EAmount := evChargeParameter.aC_EVChargeParameter.eAmount;
                    f_check_PhysicalValue_valueRange(cc_eAmount, vc_EAmount, wh,
                                                     cc_maximum_value_physicalValue_case4,
                                                     fail);
                    vc_MeterInfo.meterReading := float2int(f_CMN_PhysicalValue_GetValue(
                                                           vc_EAmount));
                    vc_EVMaxVoltage := evChargeParameter.aC_EVChargeParameter.eVMaxVoltage;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_check_PhysicalValue_valueRange(cc_eVMaxVoltage, vc_EVMaxVoltage, v,
                                             cc_maximum_value_physicalValue_case2,
                                             fail);
            vc_EVMaxCurrent := evChargeParameter.aC_EVChargeParameter.eVMaxCurrent;
            f_check_PhysicalValue_valueRange(cc_eVMaxCurrent, vc_EVMaxCurrent, a,
                                             cc_maximum_value_physicalValue_case1,
                                             fail);
            vc_EVMinCurrent := evChargeParameter.aC_EVChargeParameter.eVMinCurrent;
            f_check_PhysicalValue_valueRange(cc_eVMinCurrent, vc_EVMinCurrent, a,
                                             cc_maximum_value_physicalValue_case1,
                                             fail);
        }
        else{
            if(evChargeParameter.dC_EVChargeParameter.dC_EVStatus.eVReady == true) {
                logWarn("The field 'evReady' should be set to 'false'.");
            }
            vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                                        chargeParameterDiscoveryReq.eVChargeParameter.
                                        dC_EVChargeParameter.eVMaximumCurrentLimit;
            f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                             vc_EVMaximumCurrentLimit, a,
                                             cc_maximum_value_physicalValue_case1,
                                             fail);
            vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                                        chargeParameterDiscoveryReq.eVChargeParameter.
                                        dC_EVChargeParameter.eVMaximumVoltageLimit;
            f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                             vc_EVMaximumVoltageLimit, v,
                                             cc_maximum_value_physicalValue_case2,
                                             fail);
            if(ispresent(v_requestMessage.body.bodyElement.
                         chargeParameterDiscoveryReq.eVChargeParameter.
                         dC_EVChargeParameter.eVMaximumPowerLimit)) {
                  vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                            chargeParameterDiscoveryReq.
                                            eVChargeParameter.
                                            dC_EVChargeParameter.
                                            eVMaximumPowerLimit;
                  f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                                   vc_EVMaximumPowerLimit, w,
                                                 cc_maximum_value_physicalValue_case4,
                                                 fail);
                  vc_isPowLimit := true;
            }
            log("-------------------------------------------------------------");
            log("EV maximum charging limits:");
            log("EVMaximumCurrentLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumCurrentLimit)));
            log("EVMaximumVoltageLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumVoltageLimit)));
            if(vc_isPowLimit) {
                log("EVMaximumPowerLimit: " & toString<float>(
                f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumPowerLimit)));
            }
            log("-------------------------------------------------------------");
        }
        if(getverdict == pass) {
            setverdict(pass,"ChargeParamterDiscoveryReq message was correct. The " &
                            "previous RequestedEnergyTransferMode was selected after " &
                            "paused session.");
        }
    }
    [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                  "was misleadingly " &
                                                 "terminated by the SUT.");
    [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                     {chargeParameterDiscoveryReq := ?});
    [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                "but cannot be decoded.");
    [] a_EVCC_Unexpected_Message_Content(fail, "Received ChargeParamterDiscoveryReq " &
                                               "message, but with invalid " &
                                               "data content.");
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_EVCC_Timeout(fail);
        }
        return getverdict;
    }

    function f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_001(urn_iso_15118_2_2013_MsgDataTypes.
                                                            ResponseCodeType responseCode)
                                                            runs on EVCC_Tester
                                                            return verdicttype  {

        var charstring v_message;

          f_EVCC_setPwmMode(e_PosVolt12);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_ChargeParameterDiscoveryRes_002(
                                    responseCode, finished,
                                    m_EVCC_AC_EVSEChargeParameter_001(
                                    vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));

        tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
                                              md_CMN_BodyType_001(
                                              md_CMN_AC_PowerDeliveryReq_003(?,?,?)))) {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(fail,"Received PowerDeliveryReq message was not expected.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
        return getverdict;
    }

    function f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_002() runs on EVCC_Tester
                                                             return verdicttype  {

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                    oK, ongoing, omit,
                                    m_EVCC_AC_EVSEChargeParameter_001(
                                    vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));

        tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout - par_CMN_Transmission_Delay);

        alt {
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
                                              md_CMN_BodyType_001(
                                              md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                              *,?,?)))) {

                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass, "ChargParameterDiscoveryReq message was correct.");
                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                             md_CMN_MessageHeaderType_001(vc_SessionID),
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                        oK, ongoing, omit,
                                        m_EVCC_AC_EVSEChargeParameter_001(
                                        vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));
            repeat;
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                        "terminated by the SUT before " &
                                        "timeout has expired.");
        [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                      {chargeParameterDiscoveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received ChargeParamterDiscoveryReq " &
                                                "message, but with invalid " &
                                                "data content.");
        [] a_EVCC_Timeout(fail);
        []tc_V2G_EVCC_Ongoing_Timer.timeout{

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                        "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                                *,?,?)))) {

                    tc_V2G_SECC_Sequence_Timer.stop;
                    setverdict(pass, "ChargParameterDiscoveryReq message was correct.");
                    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
                                              md_CMN_BodyType_001(
                                              md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                              oK, ongoing, omit,
                                              m_EVCC_AC_EVSEChargeParameter_001(
                                              vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));
                    repeat;
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                        "but TCP connection " &
                                                        "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                 "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_003(in verdicttype v_vct)
                                                    runs on EVCC_Tester
                                                    return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;

    vc_AC_EVSEStatus.eVSENotification := none_;

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                md_CMN_BodyType_001(
                                md_CMN_CMN_PowerDeliveryRes_001(
                                oK, md_CMN_AC_EVSEStatus_002(vc_AC_EVSEStatus)))));

        alt {
             []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                          *,?,?)))) -> value v_requestMessage {

                  tc_V2G_SECC_Sequence_Timer.stop;
                  if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                               maxEntriesSAScheduleTuple)) {
                      vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                                      chargeParameterDiscoveryReq.
                                                      maxEntriesSAScheduleTuple;
                  }
                  var EVChargeParameter_group evChargeParameter := v_requestMessage.
                                                                   body.bodyElement.
                                                                   chargeParameterDiscoveryReq.
                                                                   eVChargeParameter;
                  vc_EAmount := evChargeParameter.aC_EVChargeParameter.eAmount;
                  f_check_PhysicalValue_valueRange(cc_eAmount, vc_EAmount, wh,
                                                   cc_maximum_value_physicalValue_case4,
                                                   v_vct);
                  vc_MeterInfo.meterReading := float2int(f_CMN_PhysicalValue_GetValue(vc_EAmount));
                  vc_EVMaxVoltage := evChargeParameter.aC_EVChargeParameter.eVMaxVoltage;
                  f_check_PhysicalValue_valueRange(cc_eVMaxVoltage, vc_EVMaxVoltage, v,
                                                   cc_maximum_value_physicalValue_case2,
                                                   v_vct);
                  vc_EVMaxCurrent := evChargeParameter.aC_EVChargeParameter.eVMaxCurrent;
                  f_check_PhysicalValue_valueRange(cc_eVMaxCurrent, vc_EVMaxCurrent, a,
                                                   cc_maximum_value_physicalValue_case1,
                                                   v_vct);
                  vc_EVMinCurrent := evChargeParameter.aC_EVChargeParameter.eVMinCurrent;
                  f_check_PhysicalValue_valueRange(cc_eVMinCurrent, vc_EVMinCurrent, a,
                                                   cc_maximum_value_physicalValue_case1,
                                                   v_vct);
                  if(ispresent(evChargeParameter.aC_EVChargeParameter.departureTime)) {
                      vc_DepartureTime := evChargeParameter.aC_EVChargeParameter.departureTime;
                      if(vc_DepartureTime == 0) {
                         logWarn("A departure time equals to 0s might lead to interoperability " &
                                 "issues if the EVSE proceeds as defined in [V2G2-303].");
                      }
                  }

                  if(getverdict == pass) {
                      setverdict(pass,"ChargeParameterDiscoveryReq message was correct. " &
                                      "Renegotiation was executed before.");

                      var SASchedules_group v_sASchedules :=
                      f_getSchedules(vc_SAScheduleTupleIdRenegotiation,
                                     vc_SAScheduleTupleIdRenegotiation2,
                                     vc_SAScheduleTupleIdRenegotiation3,
                                     vc_maxEntriesSAScheduleTuple);

                      vc_SASchedule_List := v_sASchedules.sAScheduleList;

                      if(PICS_CMN_CMN_IdentificationMode == pnC and vc_useSalesTariffs) {

                          vc_myChargeParameterBodyType := md_CMN_BodyType_001(
                                                md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                                oK, finished, v_sASchedules,
                                                m_EVCC_AC_EVSEChargeParameter_001(
                                                vc_EVSENominalVoltage, vc_EVSEMaxCurrent)));

                          vc_salesTariffIdDigestValueMap :=
                                    fx_getIdDigestMapFromScheduleListType(
                                    vc_SASchedule_List,
                                    par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
                                    - cc_salesTariff_processingTime_remains);

                          if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
                              tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                                oK, ongoing, omit,
                                                m_EVCC_AC_EVSEChargeParameter_001(
                                                vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));
                            repeat;
                        }
                    }
                }
            }
            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                          "misleadingly terminated " &
                                                          "by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                                    {chargeParameterDiscoveryReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received ChargeParamterDiscoveryReq " &
                                                        "message, but with invalid " &
                                                        "data content.");
            [] a_EVCC_Timeout(v_vct);
        }
        return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_004(HAL_61851_PwmMode_Type pwmMode)
                                                       runs on EVCC_Tester
                                                       return verdicttype  {

    var SASchedules_group v_sASchedules;
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
                v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        v_sASchedules.sAScheduleList := vc_SASchedule_List;
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                    oK, finished, v_sASchedules,
                                    m_EVCC_AC_EVSEChargeParameter_001(
                                    vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(
                                        vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_AC_PowerDeliveryReq_003(
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                start_, vc_SAScheduleTupleId, ?)))) {

                setverdict(fail,"Received PowerDeliveryReq message was not expected. " &
                                "CP State E/F should be detected before.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_005() runs on EVCC_Tester
                                                          return verdicttype  {

    var V2G_Message v_requestMessage;
    var charstring v_message;

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_AuthorizationRes_001(oK, finished))));

     alt {
          []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                        *,?,mw_EVCC_AC_EVChargeParameter_001()))))
                                        -> value v_requestMessage {

               tc_V2G_SECC_Sequence_Timer.stop;
               if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                            maxEntriesSAScheduleTuple)) {
                   vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                                   chargeParameterDiscoveryReq.
                                                   maxEntriesSAScheduleTuple;
               }
               var EVChargeParameter_group evChargeParameter := v_requestMessage.
                                                                body.bodyElement.
                                                                chargeParameterDiscoveryReq.
                                                                eVChargeParameter;

               vc_EAmount := evChargeParameter.aC_EVChargeParameter.eAmount;
               f_check_PhysicalValue_valueRange(cc_eAmount, vc_EAmount, wh,
                                                cc_maximum_value_physicalValue_case4,
                                                fail);
               vc_MeterInfo.meterReading := float2int(f_CMN_PhysicalValue_GetValue(vc_EAmount));
               vc_EVMaxVoltage := evChargeParameter.aC_EVChargeParameter.eVMaxVoltage;
               f_check_PhysicalValue_valueRange(cc_eVMaxVoltage, vc_EVMaxVoltage, v,
                                                cc_maximum_value_physicalValue_case2,
                                                fail);
               vc_EVMaxCurrent := evChargeParameter.aC_EVChargeParameter.eVMaxCurrent;
               f_check_PhysicalValue_valueRange(cc_eVMaxCurrent, vc_EVMaxCurrent, a,
                                                cc_maximum_value_physicalValue_case1,
                                                fail);
               vc_EVMinCurrent := evChargeParameter.aC_EVChargeParameter.eVMinCurrent;
               f_check_PhysicalValue_valueRange(cc_eVMinCurrent, vc_EVMinCurrent, a,
                                                cc_maximum_value_physicalValue_case1,
                                                fail);
               if(getverdict == pass) {
                   setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
               }
            }
          [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                         "was misleadingly " &
                                                        "terminated by the SUT.");
          [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {chargeParameterDiscoveryReq := ?});
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received ChargeParamterDiscoveryReq " &
                                                       "message, but with invalid " &
                                                       "data content.");
            [] a_EVCC_Timeout(fail);
        }
        return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargeParameterDiscovery_006() runs on EVCC_Tester
                                                         return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;

    vc_AC_EVSEStatus.eVSENotification := none_;

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_CMN_CMN_PowerDeliveryRes_001(
                                 oK, md_CMN_AC_EVSEStatus_002(vc_AC_EVSEStatus)))));

     alt {
            []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                          *,?,mw_EVCC_AC_EVChargeParameter_001()))))
                                          -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                             maxEntriesSAScheduleTuple)) {
                    vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                                    chargeParameterDiscoveryReq.
                                                    maxEntriesSAScheduleTuple;
                }
                var EVChargeParameter_group evChargeParameter := v_requestMessage.
                                                                 body.bodyElement.
                                                                 chargeParameterDiscoveryReq.
                                                                 eVChargeParameter;
                vc_EAmount := evChargeParameter.aC_EVChargeParameter.eAmount;
                f_check_PhysicalValue_valueRange(cc_eAmount, vc_EAmount, wh,
                                                 cc_maximum_value_physicalValue_case4,
                                                 fail);
                vc_MeterInfo.meterReading := float2int(f_CMN_PhysicalValue_GetValue(vc_EAmount));
                vc_EVMaxVoltage := evChargeParameter.aC_EVChargeParameter.eVMaxVoltage;
                f_check_PhysicalValue_valueRange(cc_eVMaxVoltage, vc_EVMaxVoltage, v,
                                                 cc_maximum_value_physicalValue_case2,
                                                 fail);
                vc_EVMaxCurrent := evChargeParameter.aC_EVChargeParameter.eVMaxCurrent;
                f_check_PhysicalValue_valueRange(cc_eVMaxCurrent, vc_EVMaxCurrent, a,
                                                 cc_maximum_value_physicalValue_case1,
                                                 fail);
                vc_EVMinCurrent := evChargeParameter.aC_EVChargeParameter.eVMinCurrent;
                f_check_PhysicalValue_valueRange(cc_eVMinCurrent, vc_EVMinCurrent, a,
                                                 cc_maximum_value_physicalValue_case1,
                                                 fail);
                if(getverdict == pass) {
                    setverdict(pass,"ChargeParameterDiscoveryReq message was correct. " &
                                    "Renegotiation was executed before.");
                }
            }
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                          "misleadingly terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                                   {chargeParameterDiscoveryReq := ?});
            [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received ChargeParamterDiscoveryReq " &
                                                       "message, but with invalid " &
                                                       "data content.");
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_EVCC_Timeout(fail);
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_001(urn_iso_15118_2_2013_MsgDataTypes.
                                                            ResponseCodeType responseCode)
                                                            runs on EVCC_Tester return verdicttype  {

        var charstring v_message;

          f_EVCC_setPwmMode(e_PosVolt12);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_ChargeParameterDiscoveryRes_002(
                                    responseCode, finished,
                                    m_EVCC_DC_EVSEChargeParameter_001()))));

        tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_CMN_DC_PreChargeReq_001(?,?,?)))) {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(fail,"Received PreChargeReq message was not expected.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_002() runs on EVCC_Tester
                                                             return verdicttype  {
        log("-------------------------------------------------------------");
        log("EVSE maximum charging limits:");
        log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumCurrentLimit)));
        log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumVoltageLimit)));
        log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumPowerLimit)));
        log("EVSEMinimumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMinimumCurrentLimit)));
        log("EVSEMinimumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMinimumVoltageLimit)));
        log("-------------------------------------------------------------");

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                    oK, ongoing, omit,
                                    md_EVCC_DC_EVSEChargeParameter_002(
                                    vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                    vc_EVSEMaximumPowerLimit,
                                    vc_EVSEMaximumVoltageLimit,
                                    vc_EVSEMinimumCurrentLimit,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                vc_EVSEMinimumVoltageLimit,
                                vc_EVSECurrentRegulationTolerance,
                                vc_EVSEPeakCurrentRipple,
                                vc_EVSEEnergyToBeDelivered)))));

tc_V2G_EVCC_Ongoing_Timer.start(par_V2G_EVCC_Ongoing_Timeout - par_CMN_Transmission_Delay);

alt {
    []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                     *,?,?)))) {

        tc_V2G_SECC_Sequence_Timer.stop;
        setverdict(pass, "ChargParameterDiscoveryReq message was correct.");
        log("-------------------------------------------------------------");
        log("EVSE maximum charging limits:");
        log("EVSEMaximumCurrentLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumCurrentLimit)));
        log("EVSEMaximumVoltageLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumVoltageLimit)));
        log("EVSEMaximumPowerLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumPowerLimit)));
        log("EVSEMinimumCurrentLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMinimumCurrentLimit)));
        log("EVSEMinimumVoltageLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMinimumVoltageLimit)));
        log("-------------------------------------------------------------");
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                oK, ongoing, omit,
                                md_EVCC_DC_EVSEChargeParameter_002(
                                vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                vc_EVSEMaximumPowerLimit,
                                vc_EVSEMaximumVoltageLimit,
                                vc_EVSEMinimumCurrentLimit,
                                vc_EVSEMinimumVoltageLimit,
                                vc_EVSECurrentRegulationTolerance,
                                vc_EVSEPeakCurrentRipple,
                                vc_EVSEEnergyToBeDelivered)))));
        repeat;
    }
    [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                  "terminated by the SUT before " &
                                                  "timeout has expired.");
    [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                {chargeParameterDiscoveryReq := ?});
    [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                "but cannot be decoded.");
    [] a_EVCC_Unexpected_Message_Content(fail, "Received ChargeParamterDiscoveryReq " &
                                               "message, but with invalid " &
                                               "data content.");
    [] a_EVCC_Timeout(fail);
    []tc_V2G_EVCC_Ongoing_Timer.timeout{

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                                *,?,?)))) {

                    tc_V2G_SECC_Sequence_Timer.stop;
                    setverdict(pass, "ChargParameterDiscoveryReq message was correct.");
                    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                                 oK, ongoing, omit,
                                                 md_EVCC_DC_EVSEChargeParameter_002(
                                                 vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                                 vc_EVSEMaximumPowerLimit,
                                                 vc_EVSEMaximumVoltageLimit,
                                                 vc_EVSEMinimumCurrentLimit,
                                                 vc_EVSEMinimumVoltageLimit,
                                                 vc_EVSECurrentRegulationTolerance,
                                                 vc_EVSEPeakCurrentRipple,
                                                 vc_EVSEEnergyToBeDelivered)))));
                    repeat;
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                 "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
    }
    return getverdict;
 }

function f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_003(in HAL_61851_Listener
                                                       v_HAL_61851_Listener,
                                                       in verdicttype v_vct)
                                                       runs on EVCC_Tester
                                                       return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;

    vc_DC_EVSEStatus.eVSENotification := none_;

        f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_PowerDeliveryRes_001(
                                oK, md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

        f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener,
                                  par_V2G_SECC_Sequence_Timeout, v_vct);

     alt {
          [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                            *,?,?)))) -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                             maxEntriesSAScheduleTuple)) {
                    vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                                    chargeParameterDiscoveryReq.
                                                    maxEntriesSAScheduleTuple;
                }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var EVChargeParameter_group evChargeParameter := v_requestMessage.
                                                 body.bodyElement.
                                                 chargeParameterDiscoveryReq.
                                                 eVChargeParameter;
if(evChargeParameter.dC_EVChargeParameter.dC_EVStatus.eVReady == true) {
    logWarn("The field 'evReady' should be set to 'false'.");
}
vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                            chargeParameterDiscoveryReq.eVChargeParameter.
                            dC_EVChargeParameter.eVMaximumCurrentLimit;
f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                 vc_EVMaximumCurrentLimit, a,
                                 cc_maximum_value_physicalValue_case1,
                                 v_vct);
vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                            chargeParameterDiscoveryReq.eVChargeParameter.
                            dC_EVChargeParameter.eVMaximumVoltageLimit;
f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                 vc_EVMaximumVoltageLimit, v,
                                 cc_maximum_value_physicalValue_case2,
                                 v_vct);
if(ispresent(v_requestMessage.body.bodyElement.
             chargeParameterDiscoveryReq.eVChargeParameter.
             dC_EVChargeParameter.eVMaximumPowerLimit)) {
      vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                chargeParameterDiscoveryReq.eVChargeParameter.
                                dC_EVChargeParameter.eVMaximumPowerLimit;
      f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                       vc_EVMaximumPowerLimit, w,
                                     cc_maximum_value_physicalValue_case4,
                                     v_vct);
      vc_isPowLimit := true;
}
log("-------------------------------------------------------------");
log("EV maximum charging limits:");
log("EVMaximumCurrentLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumCurrentLimit)));
log("EVMaximumVoltageLimit: " & toString<float>(
f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumVoltageLimit)));
if(vc_isPowLimit) {
    log("EVMaximumPowerLimit: " & toString<float>(
    f_CMN_PhysicalValue_GetValue(
        vc_EVMaximumPowerLimit)));
}
log("-------------------------------------------------------------");
if(ispresent(evChargeParameter.dC_EVChargeParameter.departureTime)) {
    vc_DepartureTime := evChargeParameter.dC_EVChargeParameter.departureTime;
    if(vc_DepartureTime == 0) {
       logWarn("A departure time equals to 0s might lead to interoperability " &
               "issues if the EVSE proceeds as defined in [V2G2-303].");
    }
}
if(getverdict == pass) {
    setverdict(pass,"ChargeParameterDiscoveryReq message was correct. " &
                    "Renegotiation was executed before.");

    var SASchedules_group v_sASchedules :=
    f_getSchedules(vc_SAScheduleTupleIdRenegotiation,
                   vc_SAScheduleTupleIdRenegotiation2,
                   vc_SAScheduleTupleIdRenegotiation3,
                   vc_maxEntriesSAScheduleTuple);

    vc_SASchedule_List := v_sASchedules.sAScheduleList;

    if(PICS_CMN_CMN_IdentificationMode == pnC and vc_useSalesTariffs) {

        log("-------------------------------------------------------------");
        log("EVSE maximum charging limits:");
        log("EVSEMaximumCurrentLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumCurrentLimit)));
        log("EVSEMaximumVoltageLimit: " & toString<float>(
        f_CMN_PhysicalValue_GetValue(
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                            vc_EVSEMaximumVoltageLimit)));
                        log("EVSEMaximumPowerLimit: " & toString<float>(
                        f_CMN_PhysicalValue_GetValue(
                            vc_EVSEMaximumPowerLimit)));
                        log("EVSEMinimumCurrentLimit: " & toString<float>(
                        f_CMN_PhysicalValue_GetValue(
                            vc_EVSEMinimumCurrentLimit)));
                        log("EVSEMinimumVoltageLimit: " & toString<float>(
                        f_CMN_PhysicalValue_GetValue(
                            vc_EVSEMinimumVoltageLimit)));
                        log("------------------------------------------------------------");

                        vc_myChargeParameterBodyType := md_CMN_BodyType_001(
                                            md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                            oK, finished, v_sASchedules,
                                            md_EVCC_DC_EVSEChargeParameter_002(
                                            vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                            vc_EVSEMaximumPowerLimit,
                                            vc_EVSEMaximumVoltageLimit,
                                            vc_EVSEMinimumCurrentLimit,
                                            vc_EVSEMinimumVoltageLimit,
                                            vc_EVSECurrentRegulationTolerance,
                                            vc_EVSEPeakCurrentRipple,
                                            vc_EVSEEnergyToBeDelivered)));

                        vc_salesTariffIdDigestValueMap :=
                                  fx_getIdDigestMapFromScheduleListType(
                                  vc_SASchedule_List,
                                  par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
                                  - cc_salesTariff_processingTime_remains);

                        if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
                            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                                oK, ongoing, omit,
                                                md_EVCC_DC_EVSEChargeParameter_002(
                                                vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                                vc_EVSEMaximumPowerLimit,
                                                vc_EVSEMaximumVoltageLimit,
                                                vc_EVSEMinimumCurrentLimit,
                                                vc_EVSEMinimumVoltageLimit,
                                                omit,
                                                vc_EVSEPeakCurrentRipple,
                                                omit)))));
                            repeat;
                        }
                    }
                }
            }
            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                           "was misleadingly " &
                                                           "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {chargeParameterDiscoveryReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received ChargeParamterDiscoveryReq " &
                                                        "message, but with invalid " &
                                                        "data content.");
            [] a_EVCC_Timeout(v_vct);
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_004(HAL_61851_PwmMode_Type pwmMode)
                                                        runs on EVCC_Tester
                                                        return verdicttype  {
        var SASchedules_group v_sASchedules;
        var boolean v_tcpTerminated := false;
        timer t_wait;

            f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
              v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        v_sASchedules.sAScheduleList := vc_SASchedule_List;
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                  oK, finished, v_sASchedules,
                                  md_EVCC_DC_EVSEChargeParameter_002(
                                  vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                  vc_EVSEMaximumPowerLimit,vc_EVSEMaximumVoltageLimit,
                                  vc_EVSEMinimumCurrentLimit,
                                  vc_EVSEMinimumVoltageLimit,
                                  vc_EVSECurrentRegulationTolerance,
                                  vc_EVSEPeakCurrentRipple,
                                  vc_EVSEEnergyToBeDelivered)))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(md_CMN_DC_CableCheckReq_001(
                                        md_CMN_DC_EVStatus_001(?,?,?))))) {

               setverdict(fail,"Received CableCheckReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_005() runs on EVCC_Tester
                                                          return verdicttype  {

    var V2G_Message v_requestMessage;
    var charstring v_message;

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_CMN_CMN_AuthorizationRes_001(oK, finished))));

     alt {
          []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                        *,?,mw_EVCC_DC_EVChargeParameter_001()))))
                                        -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                             maxEntriesSAScheduleTuple)) {
                    vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                                    chargeParameterDiscoveryReq.
                                                    maxEntriesSAScheduleTuple;
                }
                var EVChargeParameter_group evChargeParameter := v_requestMessage.
                                                                 body.bodyElement.
                                                                 chargeParameterDiscoveryReq.
                                                                 eVChargeParameter;

                if(evChargeParameter.dC_EVChargeParameter.dC_EVStatus.eVReady == true) {
                    logWarn("The field 'evReady' should be set to 'false'.");
                }
                vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                                            chargeParameterDiscoveryReq.eVChargeParameter.
                                            dC_EVChargeParameter.eVMaximumCurrentLimit;
                f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                                 vc_EVMaximumCurrentLimit, a,
                                                 cc_maximum_value_physicalValue_case1,
                                                 fail);
                vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                                            chargeParameterDiscoveryReq.eVChargeParameter.
                                            dC_EVChargeParameter.eVMaximumVoltageLimit;
                f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                                 vc_EVMaximumVoltageLimit, v,
                                                 cc_maximum_value_physicalValue_case2,
                                                 fail);
                if(ispresent(v_requestMessage.body.bodyElement.
                             chargeParameterDiscoveryReq.eVChargeParameter.
                             dC_EVChargeParameter.eVMaximumPowerLimit)) {
                      vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                                chargeParameterDiscoveryReq.eVChargeParameter.
                                                dC_EVChargeParameter.eVMaximumPowerLimit;
                      f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                                       vc_EVMaximumPowerLimit, w,
                                                     cc_maximum_value_physicalValue_case4,
                                                     fail);
                      vc_isPowLimit := true;
                }
                log("-------------------------------------------------------------");
                log("EV maximum charging limits:");
                log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                    vc_EVMaximumCurrentLimit)));
                log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                    vc_EVMaximumVoltageLimit)));
                if(vc_isPowLimit) {
                    log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                        vc_EVMaximumPowerLimit)));
                }
                log("-------------------------------------------------------------");
                if(getverdict == pass) {
                    setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
                }
            }
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                      "was misleadingly " &
                                                    "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                             {chargeParameterDiscoveryReq := ?});
            [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                  "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received ChargeParamterDiscoveryReq " &
                                                 "message, but with invalid " &
                                                 "data content.");
            [] a_EVCC_Timeout(fail);
        }
        return getverdict;
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
function f_EVCC_DC_TB_VTB_ChargeParameterDiscovery_006(in HAL_61851_Listener
                                                          v_HAL_61851_Listener)
                                                          runs on EVCC_Tester
                                                       return verdicttype  {

    var V2G_Message v_requestMessage;
    var charstring v_message;

    vc_DC_EVSEStatus.eVSENotification := none_;

        f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_PowerDeliveryRes_001(
                                oK, md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

        f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener,
                              par_V2G_SECC_Sequence_Timeout, fail);

     alt {
          []pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                          *,?,mw_EVCC_DC_EVChargeParameter_001()))))
                                          -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                             maxEntriesSAScheduleTuple)) {
                    vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                                    chargeParameterDiscoveryReq.
                                                    maxEntriesSAScheduleTuple;
                }
                var EVChargeParameter_group evChargeParameter := v_requestMessage.
                                                                 body.bodyElement.
                                                                 chargeParameterDiscoveryReq.
                                                                 eVChargeParameter;
                if(evChargeParameter.dC_EVChargeParameter.dC_EVStatus.eVReady == true) {
                    logWarn("The field 'evReady' should be set to 'false'.");
                }
                vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                                            chargeParameterDiscoveryReq.eVChargeParameter.
                                            dC_EVChargeParameter.eVMaximumCurrentLimit;
                f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                                 vc_EVMaximumCurrentLimit, a,
                                              cc_maximum_value_physicalValue_case1,
                                              fail);
                vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                                            chargeParameterDiscoveryReq.eVChargeParameter.
                                            dC_EVChargeParameter.eVMaximumVoltageLimit;
                f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                                 vc_EVMaximumVoltageLimit, v,
                                              cc_maximum_value_physicalValue_case2,
                                              fail);
                if(ispresent(v_requestMessage.body.bodyElement.
                             chargeParameterDiscoveryReq.eVChargeParameter.
                             dC_EVChargeParameter.eVMaximumPowerLimit)) {
                      vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                                chargeParameterDiscoveryReq.eVChargeParameter.
                                                dC_EVChargeParameter.eVMaximumPowerLimit;
                      f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                                       vc_EVMaximumPowerLimit, w,
                                                    cc_maximum_value_physicalValue_case4,
                                                    fail);
                      vc_isPowLimit := true;
                }
                log("-------------------------------------------------------------");
                log("EV maximum charging limits:");
                log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                    vc_EVMaximumCurrentLimit)));
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                    vc_EVMaximumVoltageLimit)));
                if(vc_isPowLimit) {
                    log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                        vc_EVMaximumPowerLimit)));
                }
                log("------------------------------------------------------------");
                if(getverdict == pass) {
                    setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
                }
            }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                       "was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {chargeParameterDiscoveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received ChargeParamterDiscoveryReq " &
                                                   "message, but with invalid " &
                                                   "data content.");
        [] a_EVCC_Timeout(fail);
    }
    return getverdict;
}

function f_checkEVChargeParameter(in PhysicalValueType vc_EVSEMaximumCurrentLimit,
                                  in PhysicalValueType vc_EVSEMaximumVoltageLimit,
                               in    PhysicalValueType vc_EVSEMinimumCurrentLimit,
                               in PhysicalValueType vc_EVSEMinimumVoltageLimit,
                               in    PhysicalValueType vc_EVMaximumCurrentLimit,
                               in PhysicalValueType vc_EVMaximumVoltageLimit,
                               in verdicttype v_vct) {

    if((f_CMN_PhysicalValue_GetValue(vc_EVMaximumCurrentLimit) <
        f_CMN_PhysicalValue_GetValue(vc_EVSEMinimumCurrentLimit)) or
       (f_CMN_PhysicalValue_GetValue(vc_EVMaximumVoltageLimit) <
        f_CMN_PhysicalValue_GetValue(vc_EVSEMinimumVoltageLimit))) {

            setverdict(v_vct, "Invalid EV charge parameter. The EVSE does not " &
                              "support the EV maximum limits.");
    }
}

function f_getSchedules(in SAIDType v_SAScheduleTupleId1,
                           in SAIDType v_SAScheduleTupleId2,
                           in SAIDType v_SAScheduleTupleId3,
                        in XSDAUX.unsignedShort v_maxEntriesSAScheduleTuple)
                           runs on EVCC_Tester return SASchedules_group {

 var SASchedules_group v_sASchedules;
 var StartType v_start;
 var DurationType v_duration;
 var StartList v_startList;
 var PmaxList v_pMaxList;
 var PhysicalValueType v_pMax;
 var PhysicalValueType v_pMaxHalved;
 var boolean v_maxEntriesChanged := false;
 var float v_processingDummy := 300.0;
 var charstring v_powerUnit := "w";
 var float v_halved := 2.0;
 var float v_third := 3.0;
 var float v_pMaxDif := 11.0;
 var charstring v_cnt1 := "1";
 var charstring v_cnt2 := "2";
 var charstring v_cnt3 := "3";

 if(isbound(vc_DepartureTime)){
     if(vc_DepartureTime > vc_maxStartValue) {
         vc_DepartureTime := cc_chargingTime24h;
         logWarn("Very high departure time value used. " &
                 "The departure time value will be " &
                 "restricted to 24h.");
     }
     if(vc_DepartureTime > cc_chargingTime24h) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            v_duration := cc_chargingTime24h;
        } else {
            v_duration := vc_DepartureTime;
        }
    }
    else {
        v_duration := cc_chargingTime24h;
    }
    if(isbound(v_maxEntriesSAScheduleTuple)) {
        if(v_maxEntriesSAScheduleTuple < vc_numberOfScheduleEntries){
            v_maxEntriesChanged := true;
        }
    }


    var float v_pMaxFloat := f_getMaxPower();
    v_pMax := fx_setPhysicalValueType(v_pMaxFloat, v_powerUnit);
    v_pMaxHalved := fx_setPhysicalValueType(fx_floor((v_pMaxFloat/v_halved),
                                        cc_decimalPoints), v_powerUnit);
    if(not v_maxEntriesChanged){
        if(isbound(vc_DepartureTime)){
            v_start := vc_DepartureTime/vc_numberOfScheduleEntries;
            if(v_start > cc_chargingTime24h) {
                v_duration := cc_chargingTime24h;
            } else {
                v_duration := v_start;
            }
        }
        else {
            v_start := cc_chargingTime24h/12;
            v_duration := v_start;
        }
        for (var integer i:=0; i<vc_numberOfScheduleEntries; i:=i + 1) {
            v_startList[i] := v_start * i;
        }
        var float v_pMaxStepSize := fx_floor(((f_CMN_PhysicalValue_GetValue(v_pMax) -
                                            f_CMN_PhysicalValue_GetValue(v_pMaxHalved))/
                                            v_pMaxDif), cc_decimalPoints);
        v_pMaxList[0] := v_pMax;
        for (var integer i:=1; i<vc_numberOfScheduleEntries-1; i:=i + 1) {
            v_pMaxList[i] := fx_setPhysicalValueType((
                            f_CMN_PhysicalValue_GetValue(
                            v_pMaxList[i-1]) - v_pMaxStepSize),
                            v_powerUnit);
        }
        v_pMaxList[vc_numberOfScheduleEntries-1] := v_pMaxHalved;
    }
    if(PICS_EVCC_CMN_PmaxSchedulewithZeroPow == none_) {

        if(vc_useSalesTariffs) {

            if(PICS_CMN_CMN_IdentificationMode == pnC) {

                if(not v_maxEntriesChanged) {

                    v_sASchedules := md_EVCC_CMN_SASchedules_001(
                            v_startList, v_duration, v_SAScheduleTupleId1,
                            v_SAScheduleTupleId2, v_SAScheduleTupleId3,
                            md_EVCC_CMN_Salestariff_001(vc_salesTariffIdName & v_cnt1,
                            v_startList, v_duration, vc_SalesTariffId, 3),
                            md_EVCC_CMN_Salestariff_001(vc_salesTariffIdName & v_cnt2,
                            v_startList, v_duration, vc_SalesTariffId2, 2),
                            md_EVCC_CMN_Salestariff_001(vc_salesTariffIdName & v_cnt3,
                            v_startList, v_duration, vc_SalesTariffId3, 1),
                            v_pMaxList);
                } else {

                    v_sASchedules := md_EVCC_CMN_SASchedules_007(
                            v_duration, v_SAScheduleTupleId1,
                            v_SAScheduleTupleId2, v_SAScheduleTupleId3,
                            md_EVCC_CMN_Salestariff_002(vc_salesTariffIdName & v_cnt1,
                            v_duration, vc_SalesTariffId, 3),
                            md_EVCC_CMN_Salestariff_002(vc_salesTariffIdName & v_cnt2,
                            v_duration, vc_SalesTariffId2, 2),
                            md_EVCC_CMN_Salestariff_002(vc_salesTariffIdName & v_cnt3,
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        v_duration, vc_SalesTariffId3, 1),
                        v_pMax);
                }

            } else {

                if(not v_maxEntriesChanged) {

                    v_sASchedules := md_EVCC_CMN_SASchedules_001(
                            v_startList, v_duration, v_SAScheduleTupleId1,
                            v_SAScheduleTupleId2, v_SAScheduleTupleId3,
                            md_EVCC_CMN_Salestariff_001(omit,
                            v_startList, v_duration, vc_SalesTariffId, 3),
                            md_EVCC_CMN_Salestariff_001(omit,
                            v_startList, v_duration, vc_SalesTariffId2, 2),
                            md_EVCC_CMN_Salestariff_001(omit,
                            v_startList, v_duration, vc_SalesTariffId3, 1),
                            v_pMaxList);
                } else {

                    v_sASchedules := md_EVCC_CMN_SASchedules_007(
                            v_duration, v_SAScheduleTupleId1,
                            v_SAScheduleTupleId2, v_SAScheduleTupleId3,
                            md_EVCC_CMN_Salestariff_002(omit,
                            v_duration, vc_SalesTariffId, 3),
                            md_EVCC_CMN_Salestariff_002(omit,
                            v_duration, vc_SalesTariffId2, 2),
                            md_EVCC_CMN_Salestariff_002(omit,
                            v_duration, vc_SalesTariffId3, 1),
                            v_pMax);
                }
            }

        } else {

            if(not v_maxEntriesChanged) {

                v_sASchedules := md_EVCC_CMN_SASchedules_004(
                                v_SAScheduleTupleId1, v_startList,
                                v_duration, v_pMaxList);
            } else {

                v_sASchedules := md_EVCC_CMN_SASchedules_008(
                     v_SAScheduleTupleId1, v_duration, v_pMax);
            }

        }

    } else if(PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge or
              (PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge and
              vc_renegotiation)) {

        if(not v_maxEntriesChanged) {

            if(vc_useSalesTariffs) {

                if(PICS_CMN_CMN_IdentificationMode == pnC) {

                    v_sASchedules := md_EVCC_CMN_SASchedules_002(
                             v_startList, (float2int(par_SECC_Pmax0W)),
                             v_duration, v_SAScheduleTupleId1,
                             v_SAScheduleTupleId2, v_SAScheduleTupleId3,
                             md_EVCC_CMN_Salestariff_001(vc_salesTariffIdName & v_cnt1,
                             v_startList, v_duration, vc_SalesTariffId, 3),
                             md_EVCC_CMN_Salestariff_001(vc_salesTariffIdName & v_cnt2,
                             v_startList, v_duration, vc_SalesTariffId2, 2),
                             md_EVCC_CMN_Salestariff_001(vc_salesTariffIdName & v_cnt3,
                             v_startList, v_duration, vc_SalesTariffId3, 1),
                             v_pMaxList);

                } else {

                    v_sASchedules := md_EVCC_CMN_SASchedules_002(
                            v_startList, (float2int(par_SECC_Pmax0W)),
                            v_duration, v_SAScheduleTupleId1,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        v_SAScheduleTupleId2, v_SAScheduleTupleId3,
                        md_EVCC_CMN_Salestariff_001(omit,
                        v_startList, v_duration, vc_SalesTariffId, 3),
                        md_EVCC_CMN_Salestariff_001(omit,
                        v_startList, v_duration, vc_SalesTariffId2, 2),
                        md_EVCC_CMN_Salestariff_001(omit,
                        v_startList, v_duration, vc_SalesTariffId3, 1),
                        v_pMaxList);
                }

            } else {

                v_sASchedules := md_EVCC_CMN_SASchedules_005(
                                 v_SAScheduleTupleId1, v_startList,
                                 (float2int(par_SECC_Pmax0W)),
                                 v_duration, v_pMaxList);

            }

            if(float2int(par_SECC_Pmax0W) > v_start) {
                setverdict(inconc, "Invalid Pmax0W time was configured. Decrease " &
                                   "'par_SECC_Pmax0W' or increase EV departure " &
                                   "time for the next test run.");
            }

        } else {
            setverdict(inconc, "PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge " &
                               "only supported if the maximal number of SASchedule entries " &
                               "has not changed by the SUT.");
            // set dummy profile
            v_sASchedules := md_EVCC_CMN_SASchedules_008(1,0, v_pMax);
        }

    } else if(PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge) {

        if(not v_maxEntriesChanged) {

            if(vc_useSalesTariffs) {

                if(PICS_CMN_CMN_IdentificationMode == pnC) {

                    v_sASchedules := md_EVCC_CMN_SASchedules_003(
                             v_startList, (float2int(par_SECC_Pmax0W) + v_start),
                             v_duration, v_SAScheduleTupleId1,
                             v_SAScheduleTupleId2, v_SAScheduleTupleId3,
                             md_EVCC_CMN_Salestariff_001(vc_salesTariffIdName & v_cnt1,
                             v_startList, v_duration, vc_SalesTariffId, 3),
                             md_EVCC_CMN_Salestariff_001(vc_salesTariffIdName & v_cnt2,
                             v_startList, v_duration, vc_SalesTariffId2, 2),
                             md_EVCC_CMN_Salestariff_001(vc_salesTariffIdName & v_cnt3,
                             v_startList, v_duration, vc_SalesTariffId3, 1),
                             v_pMaxList);

                } else {

                    v_sASchedules := md_EVCC_CMN_SASchedules_003(
                        v_startList, (float2int(par_SECC_Pmax0W) + v_start),
                        v_duration, v_SAScheduleTupleId1,
                        v_SAScheduleTupleId2, v_SAScheduleTupleId3,
                        md_EVCC_CMN_Salestariff_001(omit,
                        v_startList, v_duration, vc_SalesTariffId, 3),
                        md_EVCC_CMN_Salestariff_001(omit,
                        v_startList, v_duration, vc_SalesTariffId2, 2),
                        md_EVCC_CMN_Salestariff_001(omit,
                        v_startList, v_duration, vc_SalesTariffId3, 1),
                        v_pMaxList);

                }

            } else {

                v_sASchedules := md_EVCC_CMN_SASchedules_006(
                                 v_SAScheduleTupleId1, v_startList,
                                 (float2int(par_SECC_Pmax0W) + v_start),
                                 v_duration, v_pMaxList);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }

            if(float2int(par_SECC_Pmax0W) > v_start) {
                setverdict(inconc, "Invalid Pmax0W time was configured. Decrease " &
                                   "'par_SECC_Pmax0W' or increase EV departure time " &
                                   "for the next test run.");
            }

        } else {
            setverdict(inconc, "PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepAfterCharge " &
                               "only supported if the maximal number of SASchedule entries " &
                               "has not changed by the SUT.");
            // set dummy profile
            v_sASchedules := md_EVCC_CMN_SASchedules_008(1,0, v_pMax);
        }
         }
      tc_CMN_ScheduleListActive_Timer.start(v_processingDummy);
         return v_sASchedules;
   }

   function f_getMaxPower() runs on EVCC_Tester return float {

         var float v_pMax;

         if(PICS_CMN_CMN_ChargingMode == aC) {

          if(vc_requestedEnergyTransferMode == aC_three_phase_core) {
             v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
                       f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent) * 3.0;
          }
          else {
             v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSENominalVoltage) *
                       f_CMN_PhysicalValue_GetValue(vc_EVSEMaxCurrent);
          }
         } else {
           v_pMax := f_CMN_PhysicalValue_GetValue(vc_EVSEMaximumPowerLimit);
         }

         return v_pMax;
   }
}
```

I.3.2.12 EVCC functions for PowerDelivery

```
module TestBehavior_EVCC_PowerDelivery {

  import from Pics_15118 all;
  import from Services_HAL_61851_1 all;
  import from LibFunctions_15118_2 all;
  import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
  import from Templates_CMN_V2GMessage all;
  import from Templates_EVCC_MeteringReceipt all;
  import from Templates_CMN_PreCharge all;
  import from Templates_CMN_ChargingStatus all;
  import from Templates_CMN_CableCheck all;
  import from Templates_EVCC_ChargeParameterDiscovery all;
  import from Templates_CMN_CurrentDemand all;
  import from Templates_CMN_PowerDelivery all;
  import from Templates_CMN_SessionStop all;
  import from Templates_CMN_SessionStop all;
  import from Templates_EVCC_PaymentServiceSelection all;
  import from TestBehavior_EVCC_CommonBehavior all;
  import from Templates_CMN_PreCharge all;
  import from Pics_15118_2 all;
  import from ComponentsAndPorts all;
  import from Timer_15118_2 all;
  import from DataStructure_HAL_61851 all;
  import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
  import from Templates_CMN_V2GTcpTlsPortControl all;
  import from DataStructure_HAL_61851 all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Timer_15118 all;
import from Pixit_15118 all;
import from Pixit_15118_2 all;
import from urn_iso_15118_2_2013_MsgHeader all;
import from http_www_w3_org_2000_09_xmldsig all;
import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
import from Services_Security all;
import from urn_iso_15118_2_2013_X509Def all;
import from Templates_CMN_Security all;
import from DataStructure_V2G all;
import from TTlibrary_Logging all;

function f_EVCC_StateChange_B2CorD(in HAL_61851_Listener v_HAL_61851_Listener,
                                   in float stateTime, in verdicttype v_vct)
                                   runs on EVCC_Tester
                                   return verdicttype {
      timer stateTimer := stateTime;
      f_EVCC_confirmState(vc_state, v_HAL_61851_Listener, stateTimer, v_vct, false);
  return getverdict;
}

function f_EVCC_StateChange_CorD2B(in HAL_61851_Listener v_HAL_61851_Listener,
                                   in float stateTime, in verdicttype v_vct)
                                   runs on EVCC_Tester
                                   return verdicttype {
      timer stateTimer := stateTime;
      f_EVCC_confirmState(e_StateB, v_HAL_61851_Listener, stateTimer, v_vct, false);
  return getverdict;
}

function f_EVCC_AC_TB_VTB_PowerDelivery_001(in HAL_61851_Listener v_HAL_61851_Listener,
                                            in verdicttype v_vct)
                                            runs on EVCC_Tester return verdicttype  {

      var charstring v_message;
      var V2G_Message v_requestMessage;
      var MessageHeaderType v_messageHeader;
      var charstring v_id;
      var SASchedules_group v_sASchedules;
      var float v_processingDummy := 5.0;
      vc_stateNotDetected := true;

      if(PICS_CMN_CMN_IdentificationMode == pnC and vc_useSalesTariffs) {

          var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
          f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                                vc_myChargeParameterBodyType,
                                                vc_salesTariffIdDigestValueMap));
          var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                           vc_privateKey);

          v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);

      } else {
          v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
      }

            f_EVCC_changeValidStateCondition(e_StateB,vc_state);

      if(getverdict == pass) {
          v_sASchedules.sAScheduleList := vc_SASchedule_List;
          tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
          pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        v_messageHeader, md_CMN_BodyType_001(
                                        md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                        oK, finished, v_sASchedules,
                                        m_EVCC_AC_EVSEChargeParameter_001(
                                        vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));

          alt {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(
                                                 vc_SessionID),
                                              md_CMN_BodyType_001(
                                              md_CMN_AC_PowerDeliveryReq_003(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    start_, ?, ?))))
                                    -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
        vc_chargingProfile := v_requestMessage.body.bodyElement.
                              powerDeliveryReq.chargingProfile;

        var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                            powerDeliveryReq.sAScheduleTupleID;

            timer t_stateTimer :=  par_V2G_SECC_CPState_Detection_Timeout +
                                   par_CMN_Transmission_Delay;
            t_stateTimer.start;
            alt {
                [] a_EVCC_StateCorD(pt_HAL_61851_Internal_Port, vc_state) {
                    t_stateTimer.stop;
                    vc_stateNotDetected := false;
                }
                [] t_stateTimer.timeout {
                   vc_stateNotDetected := true;
                          setverdict(v_vct, "The EV could not signal " &
                                            "the corresponding " &
                                            "CP state within required time.");
                }
            }

        if(PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization) {
            if(v_SAScheduleTupleId != vc_SAScheduleTupleId3) {
                    setverdict(v_vct, "Wrong SAScheduleTupleId was " &
                                      "chosen from SUT. The SalesTariff " &
                                      "with the lowest price level " &
                                      "was not considered.");
            }
        } else if(PIXIT_EVCC_CMN_ChargingProfileOptimization == none_) {
            if(v_SAScheduleTupleId != vc_SAScheduleTupleId) {
                    setverdict(v_vct, "Wrong SAScheduleTupleId was " &
                                        "chosen from SUT. The SUT did not choose " &
                                        "the default SAScheduleTuple.");
            }
        } else {
            if((v_SAScheduleTupleId == vc_SAScheduleTupleId) or
               (v_SAScheduleTupleId == vc_SAScheduleTupleId2) or
               (v_SAScheduleTupleId == vc_SAScheduleTupleId3)) {
                if(getverdict == pass) {
                    setverdict(pass, "Valid SAScheduleTupleId was " &
                                     "chosen from SUT.");
                }
            } else {
                  setverdict(v_vct, "Invalid SAScheduleTupleId was " &
                                    "chosen from SUT.");
            }
        }
        vc_SAScheduleTupleId := v_SAScheduleTupleId;
        if(PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse) {
            f_checkMaxNumberOfPhasesInUse(vc_chargingProfile, v_vct);
        }

        if(getverdict == pass) {
            f_checkValidMaxPowerEntries(vc_chargingProfile, vc_SASchedule_List,
                                        vc_SAScheduleTupleId, v_vct);
        }

        if(getverdict == pass) {
            setverdict(pass,"PowerDeliveryReq message was correct.");
        }
        vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
        tc_CMN_ProcessingTime_Timer.stop;
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     md_CMN_AC_PowerDeliveryReq_003(
                                     stop_, vc_SAScheduleTupleId, *)))) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    tc_V2G_SECC_Sequence_Timer.stop;
                    if(PIXIT_EVCC_CMN_Pause == pause or
                       PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
                        vc_sleepModeWoCharging := true;
                        vc_stateNotDetected := false;
                        f_EVCC_changeValidStateCondition(e_StateB,e_StateB);
                        setverdict(pass,"PowerDeliveryReq message with ChargeProgressType " &
                                        "'Stop_' was received. The SUT wants to initiate a " &
                                        "sleep phase directly.");
                    } else {
                        setverdict(inconc,"PowerDeliveryReq message with ChargeProgressType " &
                                          "'Stop_' was received. The SUT wants to initiate a " &
                                          "sleep phase directly.");
                    }
                }
                [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                               "was misleadingly " &
                                                               "terminated by the SUT.");
                [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                                {powerDeliveryReq := ?});
                [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                             "but cannot be decoded.");
                [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq message, " &
                                                            "but with invalid data content.");
                [] a_EVCC_Timeout(v_vct);
            }
        }
        return getverdict;
}

function f_EVCC_AC_TB_VTB_PowerDelivery_002(urn_iso_15118_2_2013_MsgDataTypes.
                                            ResponseCodeType responseCode,
                                            in HAL_61851_Listener v_HAL_61851_Listener)
                                            runs on EVCC_Tester return verdicttype  {

    var charstring v_message;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

      f_EVCC_setPwmMode(e_PosVolt12);
      f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                              md_CMN_BodyType_001(
                              md_CMN_CMN_PowerDeliveryRes_001(responseCode,
                              md_CMN_AC_EVSEStatus_002(vc_AC_EVSEStatus)))));

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
      [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
         isStateB := true;
         tc_EVCC_StateB_Shutdown_Timer.stop;
         if(isClosedTCP == false) {repeat;}
      }
      [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                    "terminated by the SUT.") {
         isClosedTCP := true;
         tc_CMN_TCP_Connection_Termination_Timer.stop;
         if((isStateB == false)) {repeat;}
      }
      [] a_EVCC_CMN_SessionStop_Termination() {
          tc_CMN_TCP_Connection_Termination_Timer.start(
          par_CMN_TCP_Connection_Termination_Timeout);
          repeat;
      }
      [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        m_CMN_AC_ChargingStatusReq_001))) {
          tc_V2G_SECC_Sequence_Timer.stop;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            setverdict(fail,"Received ChargingStatusReq message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout() {
           v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
           if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
        }
        [] a_EVCC_StateB_Shutdown_Timeout() {
           v_charstring := v_charstring & "SUT has not changed to CP State B " &
                           "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
           if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
        }
     }
     if(not(isStateB and isClosedTCP) and v_charstring != "") {
        setverdict(fail,v_charstring);
     }
     return getverdict;
  }

  function f_EVCC_AC_TB_VTB_PowerDelivery_003(in HAL_61851_Listener v_HAL_61851_Listener)
                                              runs on EVCC_Tester return verdicttype  {

     timer t_wait;
     var float v_timeShift := 1.0;
     var boolean isStateB := false;
     var boolean isClosedTCP := false;
     var charstring v_charstring := "";

     // Wait until SUT message timeout
     log("Proccessing time: " & toString<float>(vc_processingTime) & "s.");
     t_wait.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq -
                  v_timeShift - vc_processingTime);

     pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

     alt {
         [] t_wait.timeout;
         [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                       "terminated by the SUT " &
                                                       "before timeout has expired.");
         [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
             setverdict(fail, "No V2G message was expected.");
         }
     }

     if(getverdict == pass) {

           f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

         sleep(v_timeShift - par_CMN_Transmission_Delay);

         tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
         tc_CMN_TCP_Connection_Termination_Timer.start(
         par_CMN_TCP_Connection_Termination_Timeout);

         alt {
           [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
              isStateB := true;
              tc_EVCC_StateB_Shutdown_Timer.stop;
              if(isClosedTCP == false) {repeat;}
           }
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT.") {
              isClosedTCP := true;
              tc_CMN_TCP_Connection_Termination_Timer.stop;
              if((isStateB == false)) {repeat;}
           }
           [] a_EVCC_CMN_SessionStop_Termination() {
               tc_CMN_TCP_Connection_Termination_Timer.start(
               par_CMN_TCP_Connection_Termination_Timeout);
               repeat;
           }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
         }
         if(not(isStateB and isClosedTCP) and v_charstring != "") {
            setverdict(fail,v_charstring);
         }
      }
      return getverdict;
   }

   function f_EVCC_AC_TB_VTB_PowerDelivery_004(in HAL_61851_Listener v_HAL_61851_Listener)
                                              runs on EVCC_Tester return verdicttype {

      var boolean isStateB := false;
      var boolean isClosedTCP := false;
      var charstring v_charstring := "";

            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

      pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

      tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
      tc_CMN_TCP_Connection_Termination_Timer.start(
      par_CMN_TCP_Connection_Termination_Timeout);

      alt {
         [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
            isStateB := true;
            tc_EVCC_StateB_Shutdown_Timer.stop;
            if(isClosedTCP == false) {repeat;}
         }
         [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                       "terminated by the SUT.") {
            isClosedTCP := true;
            tc_CMN_TCP_Connection_Termination_Timer.stop;
            if((isStateB == false)) {repeat;}
         }
         [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
         }
         [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                    "was >>NOT<< terminated by the SUT");
         [] a_EVCC_TCP_Connection_Termination_Timeout() {
            v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                            "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
            if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
         }
         [] a_EVCC_StateB_Shutdown_Timeout() {
            v_charstring := v_charstring & "SUT has not changed to CP State B " &
                            "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
            if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
         }
      }
      if(not(isStateB and isClosedTCP) and v_charstring != "") {
         setverdict(fail,v_charstring);
      }
      return getverdict;
   }

   function f_EVCC_AC_TB_VTB_PowerDelivery_005(HAL_61851_PwmMode_Type pwmMode)
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        runs on EVCC_Tester
                                        return verdicttype  {
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
             [] t_wait.timeout;
             [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                           "terminated by the SUT " &
                                                           "before request message " &
                                                           "was sent by the Test System. " &
                                                           "The send operation will be " &
                                                           "canceled.") {
                 v_tcpTerminated := true;
             }
        }
    if(not v_tcpTerminated) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_CMN_PowerDeliveryRes_001(
                                  oK, md_CMN_AC_EVSEStatus_002(vc_AC_EVSEStatus)))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
               tc_CMN_TCP_Connection_Termination_Timer.start(
               par_CMN_TCP_Connection_Termination_Timeout);
               repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     m_CMN_AC_ChargingStatusReq_001))) {

               setverdict(fail,"Received ChargingStatusReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_PowerDelivery_006(in HAL_61851_Listener v_HAL_61851_Listener)
                                            runs on EVCC_Tester return verdicttype  {

       var charstring v_message;
       var V2G_Message v_requestMessage;
       var MessageHeaderType v_messageHeader;
       var charstring v_id;
       var SASchedules_group v_sASchedules;

       if(PICS_CMN_CMN_IdentificationMode == pnC and vc_useSalesTariffs) {

          var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
          f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                                vc_myChargeParameterBodyType,
                                                vc_salesTariffIdDigestValueMap));

          var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                           vc_privateKey);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);

} else {
   v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
}


v_sASchedules.sAScheduleList := vc_SASchedule_List;
tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                            v_messageHeader, md_CMN_BodyType_001(
                            md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                            oK, finished, v_sASchedules,
                            m_EVCC_AC_EVSEChargeParameter_001(
                            vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));

alt {
  [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                   md_CMN_BodyType_001(
                                   md_CMN_AC_PowerDeliveryReq_003(
                                   start_, ?, ?))))
                                   -> value v_requestMessage{

        tc_V2G_SECC_Sequence_Timer.stop;
        var ChargingProfileType v_chargingProfile := v_requestMessage.
                                                      body.bodyElement.
                                                     powerDeliveryReq.
                                                     chargingProfile;

        var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                            powerDeliveryReq.sAScheduleTupleID;


        if(PIXIT_EVCC_CMN_OldSAScheduleTupleID == unknown) {

            if(PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization) {
               if(v_SAScheduleTupleId != vc_SAScheduleTupleIdRenegotiation3) {
                   setverdict(fail, "Wrong SAScheduleTupleId was " &
                                    "chosen from SUT. The SalesTariff " &
                                    "with the lowest price level " &
                                    "was not considered.");
               }
            } else if(PIXIT_EVCC_CMN_ChargingProfileOptimization == none_) {
               if(v_SAScheduleTupleId != vc_SAScheduleTupleIdRenegotiation) {
                   setverdict(fail, "Wrong SAScheduleTupleId was " &
                                    "chosen from SUT. The SUT did not choose " &
                                    "the default SAScheduleTuple.");
               }
            } else {
               if((v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation) or
                  (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation2) or
                  (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation3)) {
                   setverdict(pass, "Valid SAScheduleTupleId was " &
                                    "chosen from SUT.");
               } else {
                   setverdict(fail, "Invalid SAScheduleTupleId was " &
                                    "chosen from SUT.");
               }
            }
        } else {
           if(v_chargingProfile != vc_chargingProfile) {
               setverdict(fail, "The received charging profile is not equal " &
                                "to the selected charging profile before " &
                                "renegotiation.");
           }
        }
        vc_SAScheduleTupleId := v_SAScheduleTupleId;
        vc_chargingProfile := v_chargingProfile;

        if(getverdict == pass) {
            f_checkValidMaxPowerEntries(vc_chargingProfile, vc_SASchedule_List,
                                        vc_SAScheduleTupleId, fail);
        }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                if(PIXIT_EVCC_AC_MaxNumberOfPhasesInUse == maxNumberOfPhasesInUse) {
                    f_checkMaxNumberOfPhasesInUse(vc_chargingProfile, fail);
                }

                if(getverdict == pass) {
                    setverdict(pass,"PowerDeliveryReq message was correct. " &
                                    "Renegotiation was executed before.");
                }
            }
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                          "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                            {powerDeliveryReq := ?});
            [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received PowerDeliveryReq message, " &
                                                       "but with invalid data content.");
            [] a_EVCC_Timeout(fail);
        }
        return getverdict;
}

function f_checkMaxNumberOfPhasesInUse(in ChargingProfileType v_ChargingProfile,
                                       in verdicttype v_vct) {

        for (var integer i:=0; i<sizeof(v_ChargingProfile.profileEntry_list); i:=i + 1) {
         if(not ispresent(v_ChargingProfile.profileEntry_list[i].
                          chargingProfileEntryMaxNumberOfPhasesInUse)) {

                setverdict(v_vct, "ChargingProfileEntryMaxNumberOfPhasesInUse " &
                                  "field was not set.");
         }
     }
}

function f_checkValidMaxPowerEntries(in ChargingProfileType v_ChargingProfile,
                                     in SAScheduleListType v_SASchedule_List,
                                     in SAIDType v_SAScheduleTupleId,
                                     in verdicttype v_vct)
                                     runs on EVCC_Tester {
       var PhysicalValueType v_pMaxCP;
       var integer v_startValueCP1;
    var integer v_startValueCP2;
    var PhysicalValueType v_pMaxSL;
    var PhysicalValueType v_pMaxSLTemp;
    var integer v_startValueSL1;
    var integer v_startValueSL2;
    var integer v_diffCP;
    var integer v_diffSL;
    var integer v_diffSLComplete := 0;
    var integer v_index;
    var integer v_count := 0;
    var boolean v_endSL := false;
    var boolean v_resetPMaxSLTemp := true;

    for (var integer i:=0; i<sizeof(v_SASchedule_List.sAScheduleTuple_list);
            i:=i + 1) {
        if(v_SASchedule_List.sAScheduleTuple_list[i].sAScheduleTupleID ==
              v_SAScheduleTupleId) {
             v_index := i;
             break;
        }
    }

    for (var integer i:=0; i<sizeof(v_ChargingProfile.profileEntry_list) - 1; i:=i + 1) {

        v_pMaxCP := v_ChargingProfile.profileEntry_list[i].chargingProfileEntryMaxPower;
        if(i == 0) {
            if(tc_CMN_ScheduleListActive_Timer.running) {
               vc_scheduleListActiveTime := tc_CMN_ScheduleListActive_Timer.read;
               tc_CMN_ScheduleListActive_Timer.stop;
            }
               v_startValueCP1 := v_ChargingProfile.profileEntry_list[i].chargingProfileEntryStart;
        }
        v_startValueCP2 := v_ChargingProfile.profileEntry_list[i+1].chargingProfileEntryStart;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            v_diffCP := v_startValueCP2 - v_startValueCP1;
            for (var integer j:=v_count; j<sizeof(v_SASchedule_List.
                                                sAScheduleTuple_list[v_index].
                                                pMaxSchedule.pMaxScheduleEntry_list); j:=j + 1) {

                v_pMaxSL := v_SASchedule_List.sAScheduleTuple_list[v_index].pMaxSchedule.
                            pMaxScheduleEntry_list[j].pMax;
                if(v_resetPMaxSLTemp) {
                    v_pMaxSLTemp := v_pMaxSL;
                    v_resetPMaxSLTemp := false;
                }
                v_startValueSL1 := v_SASchedule_List.
                                   sAScheduleTuple_list[v_index].pMaxSchedule.
                                   pMaxScheduleEntry_list[j].timeInterval.
                                   relativeTimeInterval.start_;

                if(j == 0) {
                    v_startValueSL1 := v_startValueSL1 - float2int(vc_scheduleListActiveTime);
                }

                if(j == sizeof(v_SASchedule_List.sAScheduleTuple_list[v_index].
                               pMaxSchedule.pMaxScheduleEntry_list) - 1) {
                    var integer v_duration := v_SASchedule_List.
                                              sAScheduleTuple_list[v_index].
                                              pMaxSchedule.pMaxScheduleEntry_list[j].timeInterval.
                                              relativeTimeInterval.duration;
                    v_diffSL := v_duration;
                    v_endSL := true;

                } else {
                    v_startValueSL2 := v_SASchedule_List.
                                       sAScheduleTuple_list[v_index].pMaxSchedule.
                                       pMaxScheduleEntry_list[j+1].timeInterval.
                                       relativeTimeInterval.start_;
                    v_diffSL := v_startValueSL2 - v_startValueSL1;
                }
                v_count := v_count + 1;
                if((v_diffSLComplete != 0) and
                   (f_CMN_PhysicalValue_GetValue(v_pMaxSL) >
                    f_CMN_PhysicalValue_GetValue(v_pMaxSLTemp))) {
                    v_pMaxSL := v_pMaxSLTemp;
                }
                v_diffSLComplete := v_diffSLComplete + v_diffSL;
                if(v_diffCP == v_diffSLComplete) {
                    break;
                } else if(v_diffCP < v_diffSLComplete) {
                    v_count := v_count - 1;
                    v_diffSLComplete := v_diffSLComplete - v_diffSL;
                    break;
                } else {
                    v_pMaxSLTemp := v_pMaxSL;
                    if(v_endSL) {
                            setverdict(v_vct, "The time interval of the " &
                                              "charging profile is longer " &
                                              "than the time interval of the scheduleList.");
                            i := sizeof(v_ChargingProfile.profileEntry_list) - 1;
                    }
                }
            }
            if((f_CMN_PhysicalValue_GetValue(v_pMaxCP) > f_CMN_PhysicalValue_GetValue(v_pMaxSL))) {
                    setverdict(v_vct, "ChargingProfileEntryMaxPower element " &
                                      "is greater than than the " &
                                      "limits in respective elements of the PMaxScheduleType");
                    break;

            }
            v_resetPMaxSLTemp := true;
        }
        if(getverdict == pass) {
               setverdict(pass, "The charging profile does not exceed the PMax values of all " &
                                "PMaxScheduleEntry elements.");
        }
    }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

I.3.2.13 EVCC functions for SessionStop

```
module TestBehavior_EVCC_SessionStop {

    import from ComponentsAndPorts all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from Timer_15118 all;
    import from Pics_15118 all;
    import from Services_HAL_61851_1 all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD"all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_CMN_V2GMessage all;
    import from DataStructure_V2G_TCP_TLS_Port_Control all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Templates_CMN_SessionStop all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_CMN_WeldingDetection all;
    import from Services_EVCC_PLCLinkStatus all;
    import from Pixit_15118 all;
    import from LibFunctions_15118_2 all;
    import from LibFunctions_15118_3 all;
    import from TestBehavior_EVCC_CmSetKey all;

    function f_EVCC_AC_TB_VTB_SessionStop_001(in HAL_61851_Listener v_HAL_61851_Listener,
                                              in verdicttype v_verdict1,
                                              in verdicttype v_verdict2,
                                              in verdicttype v_vct)
                                              runs on EVCC_Tester return verdicttype {

        var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_internalResponseMessage;
        var boolean isClosed := false;
        var charstring v_message;
        var V2G_Message v_requestMessage;
        var hexstring v_Nid;
        var hexstring v_Nmk;

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_PowerDeliveryRes_001(
                                    oK, md_CMN_AC_EVSEStatus_002(vc_AC_EVSEStatus)))));

       alt {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
                                              md_CMN_BodyType_001(
                                              md_CMN_CMN_SessionStopReq_001(?))))
                                              -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                vc_ChargingSession := v_requestMessage.body.bodyElement.
                                        sessionStopReq.chargingSession;
                setverdict(pass, "SessionStopReq message was correct.");

                if (vc_ChargingSession == terminate){
                   setverdict(v_verdict1, "V2G Communication Session has been finished.");
                }
                else {
                   setverdict(v_verdict2, "V2G Communication Session has been paused.");
                }

                // Explicitely send a SessionStopRes to EVCC
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_SessionStopRes_001(oK))));

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

             alt{
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_Unexpected_Message_Content(v_vct, "Received any message, " &
                                                            "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                        setverdict(v_vct,"SUT has not terminated the TCP connection " &
                                         "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
             }

             f_EVCC_setPwmMode(e_PosVolt12);
             pt_HAL_61851_Port.clear;
             tc_CMN_ProcessingTime_Timer.start(par_T_match_leave + par_CMN_Transmission_Delay);

             if(getverdict == pass and vc_ChargingSession == terminate and
                PICS_CMN_CMN_CombinedTesting) {
                 if(vc_unmapSlac) {
                    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
                 }
                 f_EVCC_getPLCLinkTermination(par_T_match_leave +
                                              par_CMN_Transmission_Delay, fail);
                 if(vc_unmapSlac) {
                     unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
                 }
             } else if(getverdict == pass and vc_ChargingSession == pause) {
                 if(vc_unmapSlac) {
                    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
                 }
                 // store Nid and Nmk
                 v_Nmk := vc_Nmk;
                 v_Nid := vc_Nid;
                 // generate new Nid and Nmk
                 vc_Nmk := f_randomHexStringGen(32);
                 vc_Nid := fx_generateNID(vc_Nmk);
                 f_EVCC_CMN_TB_VTB_CmSetKey_001();
                 if (getverdict == pass) {
                     f_EVCC_getPLCLinkTermination(par_T_match_leave, inconc);
                 }
                 if(vc_unmapSlac) {
                    unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
                 }
                 // set stored Nid and Nmk
                 vc_Nmk := v_Nmk;
                 vc_Nid := v_Nid;
             }
         }
         [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                        "terminated by the SUT.");
         [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                         {sessionStopReq := ?});
         [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                      "but cannot be decoded.");
         [] a_EVCC_Unexpected_Message_Content(v_vct, "Received SessionStopReq message, " &
                                                     "but with invalid data content.");
         [] a_EVCC_Timeout(v_vct);
     }
     return getverdict;
 }

 function f_EVCC_AC_TB_VTB_SessionStop_002(in HAL_61851_Listener v_HAL_61851_Listener,
                                           in verdicttype v_vct)
                                           runs on EVCC_Tester
                                           return verdicttype {

     // Definiton der PowerDeliveryRes Message
     var urn_iso_15118_2_2013_MsgDataTypes.EVSEStatus_group status;
     status.aC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    status.aC_EVSEStatus.eVSENotification := none_;
    status.aC_EVSEStatus.rCD := false;
    var charstring v_message;

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_PowerDeliveryRes_001(oK, status))));


   alt {
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_SessionStopReq_001(?)))) {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass, "SessionStopReq message was correct.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                       "was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {sessionStopReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received SessionStopReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_SessionStop_003(urn_iso_15118_2_2013_MsgDataTypes.
                                          ResponseCodeType responseCode)
                                          runs on EVCC_Tester return verdicttype {

    var charstring v_message;

      f_EVCC_setPwmMode(e_PosVolt12);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_SessionStopRes_001(responseCode))));

    tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }

    return getverdict;

}

function f_EVCC_AC_TB_VTB_SessionStop_004() runs on EVCC_Tester return verdicttype  {

    timer t_wait;

    // Wait until SUT message timeout
    t_wait.start(par_V2G_EVCC_Msg_Timeout_SessionStopReq -
                 par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

    alt {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_SessionStop_005() runs on EVCC_Tester return verdicttype {

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_SessionStop_001(in HAL_61851_Listener
                                          v_HAL_61851_Listener,
                                          in verdicttype v_verdict1,
                                          in verdicttype v_verdict2)
                                          runs on EVCC_Tester
                                          return verdicttype {

   var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_internalResponseMessage;
   var boolean isClosed := false;
   var charstring v_message;
   var V2G_Message v_requestMessage;
   tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
   var hexstring v_Nid;
   var hexstring v_Nmk;

          f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

   pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_PowerDeliveryRes_001(
                                oK, md_CMN_DC_EVSEStatus_002(
                                vc_DC_EVSEStatus)))));
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
 f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener,
                           par_V2G_SECC_Sequence_Timeout, fail);

alt {
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_CMN_CMN_SessionStopReq_001(?))))
                                            -> value v_requestMessage {

         tc_V2G_SECC_Sequence_Timer.stop;
         vc_ChargingSession := v_requestMessage.body.bodyElement.
                               sessionStopReq.chargingSession;
         setverdict(pass, "SessionStopReq message was correct.");

         if (vc_ChargingSession == terminate){
            setverdict(v_verdict1, "V2G Communication Session has been finished.");
         }
         else {
            setverdict(v_verdict2, "V2G Communication Session has been paused.");
         }

         // Explicitely send a SessionStopRes to EVCC
         pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     md_CMN_CMN_SessionStopRes_001(oK))));

         tc_CMN_TCP_Connection_Termination_Timer.start(
         par_CMN_TCP_Connection_Termination_Timeout);

         alt{
             [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                           "terminated by the SUT.");
             [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                        "but TCP connection " &
                                                        "was >>NOT<< terminated by the SUT");
             [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
             }
         }

         f_EVCC_setPwmMode(e_PosVolt12);
         pt_HAL_61851_Port.clear;
         tc_CMN_ProcessingTime_Timer.start(par_T_match_leave + par_CMN_Transmission_Delay);

         if(getverdict == pass and vc_ChargingSession == terminate and
             PICS_CMN_CMN_CombinedTesting) {
              if(vc_unmapSlac) {
                 map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
              }
              f_EVCC_getPLCLinkTermination(par_T_match_leave +
                                           par_CMN_Transmission_Delay, fail);
              if(vc_unmapSlac) {
                 unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
              }
         } else if(getverdict == pass and vc_ChargingSession == pause) {
              if(vc_unmapSlac) {
                 map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
              }
              // store Nid and Nmk
              v_Nmk := vc_Nmk;
              v_Nid := vc_Nid;
              // generate new Nid and Nmk
              vc_Nmk := f_randomHexStringGen(32);
              vc_Nid := fx_generateNID(vc_Nmk);
              f_EVCC_CMN_TB_VTB_CmSetKey_001();
              if (getverdict == pass) {
                  f_EVCC_getPLCLinkTermination(par_T_match_leave, inconc);
              }
              if(vc_unmapSlac) {
                 unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
              }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    // set stored Nid and Nmk
                    vc_Nmk := v_Nmk;
                    vc_Nid := v_Nid;
                }
            }
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                          "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                             {sessionStopReq := ?});
            [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received SessionStopReq or " &
                                                        "WeldingDetectionReq message, but with " &
                                                        "invalid data content.");
            [] a_EVCC_Timeout(fail);
        }
        return getverdict;
    }
}
```

I.3.2.14 EVCC functions for ChargingStatus

```
module TestBehavior_EVCC_ChargingStatus {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from XSDAUX language "XSD" all;
    import from Services_HAL_61851_1 all;
    import from Timer_15118_2  all;
    import from ComponentsAndPorts all;
    import from Templates_CMN_ChargingStatus all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_EVCC_MeteringReceipt all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_CMN_V2GMessage all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from Pics_15118 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from DataStructure_HAL_61851 all;

    import from Pixit_15118 all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118 all;

    function f_EVCC_AC_TB_VTB_ChargingStatus_001(in HAL_61851_Listener v_HAL_61851_Listener,
                                                 in verdicttype v_vct)
                                                 runs on EVCC_Tester
                                                 return verdicttype {

        var charstring v_message;
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_PowerDeliveryRes_001(
                                    oK, md_CMN_AC_EVSEStatus_002(vc_AC_EVSEStatus)))));


        alt {
             [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          m_CMN_AC_ChargingStatusReq_001))) {

                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass,"ChargingStatusReq message was correct.");
             }
             [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                            "was misleadingly " &
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                        "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {chargingStatusReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received ChargingStatusReq message, " &
                                                "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargingStatus_002(urn_iso_15118_2_2013_MsgDataTypes.
                                             ResponseCodeType responseCode,
                                             in HAL_61851_Listener v_HAL_61851_Listener)
                                             runs on EVCC_Tester return verdicttype  {

    var charstring v_message;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

      f_EVCC_setPwmMode(e_PosVolt12);
      f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_AC_ChargingStatusRes_001(
                                responseCode, vc_EvseId,vc_SAScheduleTupleId,
                                omit,omit,omit,md_CMN_AC_EVSEStatus_001(
                                true,123,none_)))));

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
           isStateB := true;
           tc_EVCC_StateB_Shutdown_Timer.stop;
           if(isClosedTCP == false) {repeat;}
        }
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
           if((isStateB == false)) {repeat;}
        }
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        m_CMN_AC_ChargingStatusReq_001))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received ChargingStatusReq message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout() {
           v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
           if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
        }
        [] a_EVCC_StateB_Shutdown_Timeout() {
           v_charstring := v_charstring & "SUT has not changed to CP State B " &
                           "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
           if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
        }
    }
    if(not(isStateB and isClosedTCP) and v_charstring != "") {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        setverdict(fail,v_charstring);
    }
    return getverdict;
}


function f_EVCC_AC_TB_VTB_ChargingStatus_003(in HAL_61851_Listener v_HAL_61851_Listener)
                                            runs on EVCC_Tester return verdicttype  {

    timer t_wait;
    var float v_timeShift := 1.0;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

    // Wait until SUT message timeout
    t_wait.start(par_V2G_EVCC_Msg_Timeout_ChargingStatusReq -
                 v_timeShift);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

          f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
               isStateB := true;
               tc_EVCC_StateB_Shutdown_Timer.stop;
               if(isClosedTCP == false) {repeat;}
            }
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if((isStateB == false)) {repeat;}
            }
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        return getverdict;
    }

    function f_EVCC_AC_TB_VTB_ChargingStatus_004(HAL_61851_PwmMode_Type pwmMode)
                                                runs on EVCC_Tester
                                                return verdicttype  {
        var boolean v_tcpTerminated := false;
        timer t_wait;

            f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
            f_EVCC_setPwmMode(pwmMode);
            t_wait.start(par_CMN_waitForNextHAL);
            alt {
               [] t_wait.timeout;
               [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                             "terminated by the SUT " &
                                                             "before request message " &
                                                             "was sent by the Test System. " &
                                                             "The send operation will be " &
                                                             "canceled.") {
                   v_tcpTerminated := true;
               }
            }
        if(not v_tcpTerminated) {
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(md_CMN_AC_ChargingStatusRes_001(
                                      oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                      md_CMN_AC_meterInfoType("Meter", 0, omit, omit, omit),
                                      false, vc_AC_EVSEStatus))));

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(
                                            m_CMN_AC_ChargingStatusReq_001))) {

                   setverdict(fail,"Received ChargingStatusReq message was not expected. " &
                                   "CP State E/F should be detected before.");
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_EVCC_AC_TB_VTB_ChargingStatus_005(in HAL_61851_Listener v_HAL_61851_Listener)
                                                runs on EVCC_Tester return verdicttype {

        var boolean isStateB := false;
        var boolean isClosedTCP := false;
        var charstring v_charstring := "";

            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                   md_CMN_BodyType_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

        tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
               isStateB := true;
               tc_EVCC_StateB_Shutdown_Timer.stop;
               if(isClosedTCP == false) {repeat;}
            }
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if((isStateB == false)) {repeat;}
            }
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }
}
```

I.3.2.15 EVCC functions for ChargingLoopAcOrPowerDelivery

```
module TestBehavior_EVCC_ChargingLoopAcOrPowerDelivery {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from XSDAUX language "XSD" all;
    import from Services_HAL_61851_1 all;
    import from Timer_15118_2 all;
    import from ComponentsAndPorts all;
    import from Templates_CMN_ChargingStatus all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_EVCC_MeteringReceipt all;
    import from Templates_CMN_V2GMessage all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from LibFunctions_15118_2 all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Pics_15118 all;
    import from Pics_15118_2 all;
    import from TTlibrary_Logging all;
    import from Templates_CMN_SessionStop all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Templates_EVCC_ChargeParameterDiscovery all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from Services_Security all;
    import from DataStructure_HAL_61851 all;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Pixit_15118 all;
import from Timer_15118 all;
import from urn_iso_15118_2_2013_MsgHeader all;
import from http_www_w3_org_2000_09_xmldsig all;
import from Services_Security all;
import from urn_iso_15118_2_2013_X509Def all;
import from Templates_CMN_Security all;
import from DataStructure_V2G all;
import from TestBehavior_EVCC_ChargeParameterDiscovery all;

function f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_001(in HAL_61851_Listener
                                                            v_HAL_61851_Listener,
                                                            in verdicttype v_vct)
                                                            runs on EVCC_Tester
                                                            return verdicttype  {

    var charstring v_message;
    var V2G_Message v_requestMessage;
    var integer v_loopCounter := 1;
    var integer v_renegotiationCounter := 0;
    var MessageHeaderType v_messageHeader;
    var charstring v_id;
    var SASchedules_group v_sASchedules;
    var boolean v_isFinished := true;
    var timer t_stopProcessing;
    vc_stateNotDetected := true;


    log("-------------------------------------------------------------");
    log("Number of received ChargingStatusReq messages: " & toString<integer>(v_loopCounter));
    log("-------------------------------------------------------------");
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    if(PICS_CMN_CMN_IdentificationMode == pnC) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_AC_ChargingStatusRes_001(
                                    oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                    vc_MeterInfo, false, vc_AC_EVSEStatus))));

    } else {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_AC_ChargingStatusRes_001(
                                    oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                    omit, false, vc_AC_EVSEStatus))));
    }

    alt {
        [not vc_renegotiation] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                        md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_AC_PowerDeliveryReq_003(
                                        stop_, vc_SAScheduleTupleId, *))))){

            timer t_stateTimer :=  par_V2G_SECC_CPState_Detection_Timeout +
                                   par_CMN_Transmission_Delay;
            t_stateTimer.start;
            alt {
              [] a_EVCC_StateB(pt_HAL_61851_Internal_Port) {
                  t_stateTimer.stop;
                  vc_stateNotDetected := false;
              }
              [] t_stateTimer.timeout {
                 vc_stateNotDetected := true;
                      setverdict(v_vct, "The EV could not signal the corresponding " &
                                        "CP state within required time.");
               }
              }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  setverdict(pass,"PowerDeliveryReq message was correct.");


  tc_V2G_SECC_Sequence_Timer.stop;
  vc_ChargeProgress := stop_;

}
[v_renegotiationCounter == 0]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_AC_PowerDeliveryReq_003(
                                renegotiate, vc_SAScheduleTupleId, *)))){

  tc_V2G_SECC_Sequence_Timer.stop;
  t_stopProcessing.stop;

  vc_renegotiation := true;
  v_renegotiationCounter := v_renegotiationCounter + 1;
  setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                 "be executed.");

  tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
  pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_PowerDeliveryRes_001(
                                oK, md_CMN_AC_EVSEStatus_002(vc_AC_EVSEStatus)))));
  vc_AC_EVSEStatus.eVSENotification := none_;
  repeat;

}
[vc_renegotiation and v_renegotiationCounter == 1]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_ChargeParamterDiscoveryReq_001(*,?,?))))
                                -> value v_requestMessage {

      v_isFinished := true;
      tc_V2G_SECC_Sequence_Timer.stop;
      if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                   maxEntriesSAScheduleTuple)) {
          vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                          chargeParameterDiscoveryReq.
                                          maxEntriesSAScheduleTuple;
      }
      v_renegotiationCounter := v_renegotiationCounter + 1;
      var EVChargeParameter_group evChargeParameter := v_requestMessage.body.
                                                       bodyElement.
                                                       chargeParameterDiscoveryReq.
                                                       eVChargeParameter;
      vc_EAmount := evChargeParameter.aC_EVChargeParameter.eAmount;
      f_check_PhysicalValue_valueRange(cc_eAmount, vc_EAmount, wh,
                                       cc_maximum_value_physicalValue_case4,
                                       v_vct);
      vc_MeterInfo.meterReading := float2int(f_CMN_PhysicalValue_GetValue(vc_EAmount));
      vc_EVMaxVoltage := evChargeParameter.aC_EVChargeParameter.eVMaxVoltage;
      f_check_PhysicalValue_valueRange(cc_eVMaxVoltage, vc_EVMaxVoltage, v,
                                       cc_maximum_value_physicalValue_case2,
                                       v_vct);
      vc_EVMaxCurrent := evChargeParameter.aC_EVChargeParameter.eVMaxCurrent;
      f_check_PhysicalValue_valueRange(cc_eVMaxCurrent, vc_EVMaxCurrent, a,
                                       cc_maximum_value_physicalValue_case1,
                                       v_vct);
      vc_EVMinCurrent := evChargeParameter.aC_EVChargeParameter.eVMinCurrent;
      f_check_PhysicalValue_valueRange(cc_eVMinCurrent, vc_EVMinCurrent, a,
                                       cc_maximum_value_physicalValue_case1,
                                       v_vct);
      if(ispresent(evChargeParameter.aC_EVChargeParameter.departureTime)) {
          vc_DepartureTime := evChargeParameter.aC_EVChargeParameter.departureTime;
          if(vc_DepartureTime == 0) {
             logWarn("A departure time equals to 0s might lead to interoperability " &
                     "issues if the EVSE proceeds as defined in [V2G2-303].");
          }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        if(getverdict == pass) {
            setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
          v_sASchedules := f_getSchedules(vc_SAScheduleTupleIdRenegotiation,
                                          vc_SAScheduleTupleIdRenegotiation2,
                                          vc_SAScheduleTupleIdRenegotiation3,
                                          vc_maxEntriesSAScheduleTuple);
          vc_SASchedule_List := v_sASchedules.sAScheduleList;

          if(PICS_CMN_CMN_IdentificationMode == pnC and vc_useSalesTariffs) {

              vc_myChargeParameterBodyType := md_CMN_BodyType_001(
                                        md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                        oK, finished, v_sASchedules,
                                        m_EVCC_AC_EVSEChargeParameter_001(
                                        vc_EVSENominalVoltage, vc_EVSEMaxCurrent)));

              vc_salesTariffIdDigestValueMap :=
                        fx_getIdDigestMapFromScheduleListType(
                        vc_SASchedule_List,
                        par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
                        - cc_salesTariff_processingTime_remains);

              if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
                  tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                  pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                        oK, ongoing, omit,
                                        m_EVCC_AC_EVSEChargeParameter_001(
                                        vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));
                  v_isFinished := false;
              } else {

                var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
                f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                                    vc_myChargeParameterBodyType,
                                                    vc_salesTariffIdDigestValueMap));
                  var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                                   vc_privateKey);

                v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);
              }
            } else {
                v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
            }

            if(v_isFinished) {
                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                 v_messageHeader,
                                               md_CMN_BodyType_001(
                                               md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                               oK, finished, v_sASchedules,
                                               m_EVCC_AC_EVSEChargeParameter_001(
                                               vc_EVSENominalVoltage,
                                               vc_EVSEMaxCurrent)))));
            }
            repeat;
        }
    }
    [vc_renegotiation and v_renegotiationCounter == 2]
    pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_AC_PowerDeliveryReq_003(
                                start_, ?, ?)))) -> value v_requestMessage{

        tc_V2G_SECC_Sequence_Timer.stop;
        var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                            powerDeliveryReq.sAScheduleTupleID;

        vc_chargingProfile := v_requestMessage.body.bodyElement.
                              powerDeliveryReq.chargingProfile;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if((v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation) or
           (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation2) or
           (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation3)) {
            setverdict(pass, "Valid SAScheduleTupleId was " &
                             "chosen from SUT.");
        } else {
                setverdict(v_vct, "Invalid SAScheduleTupleId was " &
                                  "chosen from SUT.");
        }
        vc_SAScheduleTupleId := v_SAScheduleTupleId;

        if(getverdict == pass) {
            f_checkValidMaxPowerEntries(vc_chargingProfile, vc_SASchedule_List,
                                        vc_SAScheduleTupleId, v_vct);
        }

        if(getverdict == pass) {
            vc_renegotiation := false;
            v_renegotiationCounter := 0;
            setverdict(pass,"PowerDeliveryReq message was correct.");
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(
                                              vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_CMN_CMN_PowerDeliveryRes_001(
                                            oK, md_CMN_AC_EVSEStatus_002(
                                            vc_AC_EVSEStatus)))));

            repeat;
        }
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        m_CMN_AC_ChargingStatusReq_001))) {
        tc_V2G_SECC_Sequence_Timer.stop;

        if(vc_AC_EVSEStatus.eVSENotification == stopCharging or
           vc_AC_EVSEStatus.eVSENotification == reNegotiation) {
            if(not t_stopProcessing.running) {
               t_stopProcessing.start(par_EVCC_StopCharging_Timeout);
            }
        }
        v_loopCounter := v_loopCounter + 1;
        log("-------------------------------------------------------------");
        log("Number of received ChargingStatusReq messages: " &
            toString<integer>(v_loopCounter));
        log("-------------------------------------------------------------");



        if(v_loopCounter == PICS_CMN_CMN_LoopCounter) {
            vc_AC_EVSEStatus.eVSENotification := stopCharging;
            vc_AC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
                f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);
        }
        if(v_loopCounter == PICS_CMN_CMN_RenegotiationLoopIndication and vc_renegotiation) {
             vc_AC_EVSEStatus.eVSENotification := reNegotiation;
             vc_AC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
        }
        if(getverdict == pass) {
            setverdict(pass,"ChargingStatusReq message was correct. Waiting for " &
                            "PowerDeliveryReq message");

            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            if(PICS_CMN_CMN_IdentificationMode == pnC) {
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_AC_ChargingStatusRes_001(
                                        oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                        vc_MeterInfo, false, vc_AC_EVSEStatus))));
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                } else {
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_AC_ChargingStatusRes_001(
                                        oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                        omit, false, vc_AC_EVSEStatus))));
                }
                repeat;

            }
        }
        [] t_stopProcessing.timeout {
            setverdict(fail, "SUT did not react to EVSENotification " &
                             "'stopCharging' or 'renegotiate' " &
                             "within 'par_EVCC_StopCharging_Timeout'.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                   {chargingStatusReq := ?},
                                                   {powerDeliveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                   "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                   "ChargingStatusReq message, " &
                                                   "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }

    return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_002(in HAL_61851_Listener
                                                            v_HAL_61851_Listener,
                                                            in verdicttype v_vct)
                                                            runs on EVCC_Tester
                                                            return verdicttype  {

    var charstring v_message;
    var V2G_Message v_requestMessage;
    var integer v_loopCounter := 1;
    var boolean v_isMetering := false;
    var integer v_renegotiationCounter := 0;
    var boolean isSignatureOk := true;
    var MessageHeaderType v_messageHeader;
    var charstring v_id;
    var SASchedules_group v_sASchedules;
    var boolean v_isFinished := true;
    var timer t_stopProcessing;
    vc_stateNotDetected := true;


    log("-------------------------------------------------------------");
    log("Number of received ChargingStatusReq messages: " & toString<integer>(v_loopCounter));
    log("-------------------------------------------------------------");
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                      md_CMN_MessageHeaderType_001(vc_SessionID),
                      md_CMN_BodyType_001(
                      md_CMN_AC_ChargingStatusRes_001(
                      oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                      vc_MeterInfo, true, vc_AC_EVSEStatus))));

    alt {
        [v_isMetering and not vc_renegotiation]
        pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_AC_PowerDeliveryReq_003(
                                        stop_, vc_SAScheduleTupleId, *))){

            timer t_stateTimer :=  par_V2G_SECC_CPState_Detection_Timeout +
                                   par_CMN_Transmission_Delay;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        t_stateTimer.start;
        alt {
            [] a_EVCC_StateB(pt_HAL_61851_Internal_Port) {
               t_stateTimer.stop;
               vc_stateNotDetected := false;
            }
            [] t_stateTimer.timeout {
             vc_stateNotDetected := true;
                    setverdict(v_vct, "The EV could not signal the corresponding " &
                                      "CP state within required time.");
            }
        }
      setverdict(pass,"PowerDeliveryReq message was correct.");


      tc_V2G_SECC_Sequence_Timer.stop;
      vc_ChargeProgress := stop_;
  }
  [v_isMetering and v_renegotiationCounter == 0]
  pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_AC_PowerDeliveryReq_003(
                                  renegotiate, vc_SAScheduleTupleId, *)))){
    tc_V2G_SECC_Sequence_Timer.stop;
    t_stopProcessing.stop;
    vc_renegotiation := true;
    v_renegotiationCounter := v_renegotiationCounter + 1;
    setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                    "be executed.");


    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_CMN_CMN_PowerDeliveryRes_001(
                                 oK, md_CMN_AC_EVSEStatus_002(vc_AC_EVSEStatus)))));
    vc_AC_EVSEStatus.eVSENotification := none_;
    repeat;

  }
  [vc_renegotiation and v_renegotiationCounter == 1]
  pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                  *,?,?)))) -> value v_requestMessage {

        v_isFinished := true;
        tc_V2G_SECC_Sequence_Timer.stop;
        if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                     maxEntriesSAScheduleTuple)) {
            vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                            chargeParameterDiscoveryReq.
                                            maxEntriesSAScheduleTuple;
        }
        v_renegotiationCounter := v_renegotiationCounter + 1;
        var EVChargeParameter_group evChargeParameter := v_requestMessage.body.bodyElement.
                                                         chargeParameterDiscoveryReq.
                                                         eVChargeParameter;
        vc_EAmount := evChargeParameter.aC_EVChargeParameter.eAmount;
        f_check_PhysicalValue_valueRange(cc_eAmount, vc_EAmount, wh,
                                         cc_maximum_value_physicalValue_case4,
                                         v_vct);
        vc_MeterInfo.meterReading := float2int(f_CMN_PhysicalValue_GetValue(vc_EAmount));
        vc_EVMaxVoltage := evChargeParameter.aC_EVChargeParameter.eVMaxVoltage;
        f_check_PhysicalValue_valueRange(cc_eVMaxVoltage, vc_EVMaxVoltage, v,
                                         cc_maximum_value_physicalValue_case2,
                                         v_vct);
        vc_EVMaxCurrent := evChargeParameter.aC_EVChargeParameter.eVMaxCurrent;
        f_check_PhysicalValue_valueRange(cc_eVMaxCurrent, vc_EVMaxCurrent, a,
                                         cc_maximum_value_physicalValue_case1,
                                         v_vct);
        vc_EVMinCurrent := evChargeParameter.aC_EVChargeParameter.eVMinCurrent;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        f_check_PhysicalValue_valueRange(cc_eVMinCurrent, vc_EVMinCurrent, a,
                                         cc_maximum_value_physicalValue_case1,
                                         v_vct);
        if(ispresent(evChargeParameter.aC_EVChargeParameter.departureTime)) {
            vc_DepartureTime := evChargeParameter.aC_EVChargeParameter.departureTime;
            if(vc_DepartureTime == 0) {
               logWarn("A departure time equals to 0s might lead to interoperability " &
                       "issues if the EVSE proceeds as defined in [V2G2-303].");
            }
        }
        if(getverdict == pass) {
          setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
          v_sASchedules := f_getSchedules(vc_SAScheduleTupleIdRenegotiation,
                                          vc_SAScheduleTupleIdRenegotiation2,
                                          vc_SAScheduleTupleIdRenegotiation3,
                                          vc_maxEntriesSAScheduleTuple);
          vc_SASchedule_List := v_sASchedules.sAScheduleList;

          if(vc_useSalesTariffs) {

              vc_myChargeParameterBodyType := md_CMN_BodyType_001(
                                      md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                      oK, finished, v_sASchedules,
                                      m_EVCC_AC_EVSEChargeParameter_001(
                                      vc_EVSENominalVoltage, vc_EVSEMaxCurrent)));

              vc_salesTariffIdDigestValueMap :=
                        fx_getIdDigestMapFromScheduleListType(
                        vc_SASchedule_List,
                        par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
                        - cc_salesTariff_processingTime_remains);

              if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
                  tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                  pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                      oK, ongoing, omit,
                                      m_EVCC_AC_EVSEChargeParameter_001(
                                      vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));
                  v_isFinished := false;
              } else {

                var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
                f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                                vc_myChargeParameterBodyType,
                                                vc_salesTariffIdDigestValueMap));
                var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                                 vc_privateKey);

                v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);
              }
            } else {
                v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
            }

            if(v_isFinished) {
                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        v_messageHeader,
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                        oK, finished, v_sASchedules,
                                        m_EVCC_AC_EVSEChargeParameter_001(
                                        vc_EVSENominalVoltage,
                                        vc_EVSEMaxCurrent)))));
            }
            repeat;
        }
    }
    [vc_renegotiation and v_renegotiationCounter == 2]
    pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    md_CMN_AC_PowerDeliveryReq_003(
                                    start_, ?, ?)))) -> value v_requestMessage{

        tc_V2G_SECC_Sequence_Timer.stop;
        var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                            powerDeliveryReq.sAScheduleTupleID;

        vc_chargingProfile := v_requestMessage.body.bodyElement.
                              powerDeliveryReq.chargingProfile;


        if((v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation) or
           (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation2) or
           (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation3)) {
            setverdict(pass, "Valid SAScheduleTupleId was " &
                             "chosen from SUT.");
            vc_SAScheduleTupleId := v_SAScheduleTupleId;
        } else {
                setverdict(v_vct, "Invalid SAScheduleTupleId was " &
                                  "chosen from SUT.");
        }

        if(getverdict == pass) {
            f_checkValidMaxPowerEntries(vc_chargingProfile, vc_SASchedule_List,
                                        vc_SAScheduleTupleId, v_vct);
        }

        if(getverdict == pass) {
            vc_renegotiation := false;
            v_renegotiationCounter := 0;
            setverdict(pass,"PowerDeliveryReq message was correct.");
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(
                                              vc_SessionID),
                                            md_CMN_BodyType_001(
                                            md_CMN_CMN_PowerDeliveryRes_001(
                                            oK, md_CMN_AC_EVSEStatus_002(
                                            vc_AC_EVSEStatus)))));

            repeat;
        }
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        m_CMN_AC_ChargingStatusReq_001))) {
        tc_V2G_SECC_Sequence_Timer.stop;
        if(not v_isMetering) {
            setverdict(v_vct,"The SUT did not send a Metering Receipt message.");
        }
        if(vc_AC_EVSEStatus.eVSENotification == stopCharging or
           vc_AC_EVSEStatus.eVSENotification == reNegotiation) {
            if(not t_stopProcessing.running) {
               t_stopProcessing.start(par_EVCC_StopCharging_Timeout);
            }
        }
        v_loopCounter := v_loopCounter + 1;
        log("-------------------------------------------------------------");
        log("Number of received ChargingStatusReq messages: " &
            toString<integer>(v_loopCounter));
        log("-------------------------------------------------------------");



        if(v_loopCounter == PICS_CMN_CMN_LoopCounter) {
             vc_AC_EVSEStatus.eVSENotification := stopCharging;
             vc_AC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;

                f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);
        }
        if(v_loopCounter == PICS_CMN_CMN_RenegotiationLoopIndication and
                            vc_renegotiation) {
             vc_AC_EVSEStatus.eVSENotification := reNegotiation;
             vc_AC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            if(getverdict == pass) {
                setverdict(pass,"ChargingStatusReq message was correct. Waiting " &
                                "for PowerDeliveryReq message");

                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_AC_ChargingStatusRes_001(
                                  oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                  vc_MeterInfo, true, vc_AC_EVSEStatus))));
                repeat;
            }
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_003(vc_SessionID,
                                        md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_MeteringReceiptReq_001(
                                        *, vc_SessionID, *, vc_MeterInfo))))
                                        -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            v_isMetering := true;
            var Signature authSignature := v_requestMessage.header.signature_;
            var BodyType meteringReceiptReqBodyType := v_requestMessage.body;

            f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                  cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

            if(getverdict == pass) {

                    isSignatureOk := f_verifySignature(meteringReceiptReqBodyType,
                                                       authSignature,
                                                       vc_contractSignatureCertChain.
                                                       x509Certificate, v_vct);

                if(isSignatureOk) {
                    setverdict(pass,"MeteringReceiptReq message was correct.");
                    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_AC_MeteringReceiptRes_001(
                                                oK,vc_AC_EVSEStatus))));
                    repeat;
                }
                else {
                    setverdict(v_vct, "Invalid signature of the MeteringReceiptReq message " &
                                      "was received.");
                }
            }
        }
        [] t_stopProcessing.timeout {
            setverdict(fail, "SUT did not react to EVSENotification " &
                             "'stopCharging' or 'renegotiate' " &
                             "within 'par_EVCC_StopCharging_Timeout'.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                       "was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var3(v_vct, "Unexpected V2G message was received.",
                                                   {meteringReceiptReq := ?},
                                                   {chargingStatusReq := ?},
                                                   {powerDeliveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                   "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                   "ChargingStatusReq message, " &
                                                   "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

    function f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_003(urn_iso_15118_2_2013_MsgDataTypes.
                                                                ResponseCodeType responseCode)
                                                                runs on EVCC_Tester
                                                                return verdicttype {

        var charstring v_message;

          f_EVCC_setPwmMode(e_PosVolt12);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_PowerDeliveryRes_001(
                                    responseCode, md_CMN_AC_EVSEStatus_002(vc_AC_EVSEStatus)))));

        tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
                                              md_CMN_BodyType_001(
                                              md_CMN_CMN_SessionStopReq_001(?)))) {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(fail,"Received SessionStopReq message was not expected.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
        return getverdict;
    }

    function f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_004() runs on EVCC_Tester
                                                                  return verdicttype  {

        timer t_wait;

        // Wait until SUT message timeout
        t_wait.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq -
                     par_CMN_Transmission_Delay);
        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));

        alt {
            [] t_wait.timeout;
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                          "terminated by the SUT " &
                                                          "before timeout has expired.");
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
                setverdict(fail, "No V2G message was expected.");
            }
        }

        if(getverdict == pass) {


tc_CMN_TCP_Connection_Termination_Timer.start(par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was terminated " &
                                                              "by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_005(in HAL_61851_Listener
                                                            v_HAL_61851_Listener,
                                                            in verdicttype v_vct)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

    var charstring v_message;
    var boolean v_renegotiation := false;
    var timer t_stopProcessing;


    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    if(PICS_CMN_CMN_IdentificationMode == pnC) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                            md_CMN_MessageHeaderType_001(vc_SessionID),
                            md_CMN_BodyType_001(
                            md_CMN_AC_ChargingStatusRes_001(
                            oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                            vc_MeterInfo, false, vc_AC_EVSEStatus))));
    } else {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                            md_CMN_MessageHeaderType_001(vc_SessionID),
                            md_CMN_BodyType_001(
                            md_CMN_AC_ChargingStatusRes_001(
                            oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                            omit, false, vc_AC_EVSEStatus))));
    }

    alt {
        [v_renegotiation]
        pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_AC_PowerDeliveryReq_003(
                                        renegotiate, vc_SAScheduleTupleId, *)))){
          tc_V2G_SECC_Sequence_Timer.stop;

          setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                            "be executed.");
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        m_CMN_AC_ChargingStatusReq_001))) {

            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(pass,"ChargingStatusReq message was correct. " &
                            "Waiting for PowerDeliveryReq message");
            vc_AC_EVSEStatus.eVSENotification := reNegotiation;
            vc_AC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
            v_renegotiation := true;


            if(not t_stopProcessing.running) {
                t_stopProcessing.start(par_EVCC_StopCharging_Timeout);
            }


            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if(PICS_CMN_CMN_IdentificationMode == pnC) {
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_AC_ChargingStatusRes_001(
                                oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                vc_MeterInfo, false, vc_AC_EVSEStatus))));
            } else {
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_AC_ChargingStatusRes_001(
                                oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                omit, false, vc_AC_EVSEStatus))));
            }
            if(getverdict == pass) {
               repeat;
            }
        }
        [] t_stopProcessing.timeout {
            setverdict(fail, "SUT did not react to EVSENotification 'renegotiate' " &
                             "within 'par_EVCC_StopCharging_Timeout'.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                   {chargingStatusReq := ?},
                                                   {powerDeliveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                   "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                   "ChargingStatusReq message, " &
                                                   "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_006(in HAL_61851_Listener
                                                            v_HAL_61851_Listener,
                                                            in verdicttype v_vct)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

    var charstring v_message;
    var boolean v_isMetering := false;
    var boolean v_renegotiation := false;
    var V2G_Message v_requestMessage;
    var boolean isSignatureOk := true;
    var timer t_stopProcessing;


    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_AC_MeteringReceiptRes_001(
                                oK,vc_AC_EVSEStatus))));

    alt {
        [v_renegotiation and v_isMetering]
        pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_AC_PowerDeliveryReq_003(
                                        renegotiate, vc_SAScheduleTupleId, *)))){
          tc_V2G_SECC_Sequence_Timer.stop;
          setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                          "be executed.");
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        m_CMN_AC_ChargingStatusReq_001))) {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    tc_V2G_SECC_Sequence_Timer.stop;
    setverdict(pass,"ChargingStatusReq message was correct. " &
                    "Waiting for PowerDeliveryReq message");
    vc_AC_EVSEStatus.eVSENotification := reNegotiation;
    vc_AC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;


    if(not t_stopProcessing.running) {
        t_stopProcessing.start(par_EVCC_StopCharging_Timeout);
    }


    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                     md_CMN_MessageHeaderType_001(vc_SessionID),
                     md_CMN_BodyType_001(
                     md_CMN_AC_ChargingStatusRes_001(
                     oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                     vc_MeterInfo, false, vc_AC_EVSEStatus))));
    if(getverdict == pass) {
       repeat;
    }
}
[not v_renegotiation]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_003(vc_SessionID,
                                 md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                 md_CMN_BodyType_001(
                                 md_EVCC_CMN_MeteringReceiptReq_001(
                                 *, vc_SessionID, *, vc_MeterInfo))))
                                 -> value v_requestMessage{

    tc_V2G_SECC_Sequence_Timer.stop;
    v_isMetering := true;
    v_renegotiation := true;
    var Signature authSignature := v_requestMessage.header.signature_;
    var BodyType meteringReceiptReqBodyType := v_requestMessage.body;

    f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                              cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

    if(getverdict == pass) {

            isSignatureOk := f_verifySignature(meteringReceiptReqBodyType,
                                               authSignature,
                                               vc_contractSignatureCertChain.
                                               x509Certificate, v_vct);

        if(isSignatureOk) {
            setverdict(pass,"MeteringReceiptReq message was correct.");
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_EVCC_AC_MeteringReceiptRes_001(
                                         oK,vc_AC_EVSEStatus))));
            repeat;
        }
        else {
            setverdict(v_vct, "Invalid signature of the MeteringReceiptReq " &
                              "message was received.");
        }
    }
}
[] t_stopProcessing.timeout {
    setverdict(fail, "SUT did not react to EVSENotification 'renegotiate' " &
                     "within 'par_EVCC_StopCharging_Timeout'.");
}
[] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                               "terminated by the SUT.");
[] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                            {chargingStatusReq := ?},
                                            {powerDeliveryReq := ?});
[isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                            "but cannot be decoded.");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                    "ChargingStatusReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_007() runs on EVCC_Tester
                                                              return verdicttype {

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_008(HAL_61851_PwmMode_Type pwmMode)
                                                            runs on EVCC_Tester
                                                            return verdicttype  {

    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     md_CMN_CMN_PowerDeliveryRes_001(
                                     oK, md_CMN_AC_EVSEStatus_002(vc_AC_EVSEStatus)))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_SessionStopReq_001(?)))) {

               setverdict(fail,"Received SessionStopReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_009(in HAL_61851_Listener
                                                            v_HAL_61851_Listener,
                                                            in verdicttype v_vct)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

    var charstring v_message;
    var integer v_loopCounter := 1;



    log("-------------------------------------------------------------");
    log("Number of received ChargingStatusReq messages: " &
        toString<integer>(v_loopCounter));
    log("-------------------------------------------------------------");
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                      md_CMN_MessageHeaderType_001(vc_SessionID),
                      md_CMN_BodyType_001(
                      md_CMN_AC_ChargingStatusRes_001(
                      oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                      vc_MeterInfo, false, vc_AC_EVSEStatus))));

    alt {
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_AC_PowerDeliveryReq_003(
                                      renegotiate, vc_SAScheduleTupleId, *)))){
          tc_V2G_SECC_Sequence_Timer.stop;
          setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                          "be executed.");
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      m_CMN_AC_ChargingStatusReq_001))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(pass,"ChargingStatusReq message was correct. " &
                            "Waiting for PowerDeliveryReq message");

            v_loopCounter := v_loopCounter + 1;
            log("-------------------------------------------------------------");
            log("Number of received ChargingStatusReq messages: " &
                toString<integer>(v_loopCounter));
            log("-------------------------------------------------------------");

            if(v_loopCounter == PICS_CMN_CMN_LoopCounter) {
                setverdict(inconc, "No Renegotiation was triggered by the SUT in the " &
                                   "configured number of charging loop sequences" &
                                   "(PICS_CMN_CMN_LoopCounter).");
            }
            if(getverdict == pass) {
                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_AC_ChargingStatusRes_001(
                                oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                vc_MeterInfo, false, vc_AC_EVSEStatus))));
                repeat;
            }
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                   {chargingStatusReq := ?},
                                                   {powerDeliveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                   "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                   "ChargingStatusReq message, " &
                                                   "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_ChargingLoopAcOrPowerDelivery_010(in HAL_61851_Listener
                                                            v_HAL_61851_Listener,
                                                            in verdicttype v_vct)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

    var charstring v_message;
    var boolean v_isMetering := false;
    var V2G_Message v_requestMessage;
    var boolean isSignatureOk := true;
    var integer v_loopCounter := 1;

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_AC_MeteringReceiptRes_001(
                                oK,vc_AC_EVSEStatus))));

    alt {
        [v_isMetering] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_AC_PowerDeliveryReq_003(
                                        renegotiate, vc_SAScheduleTupleId, *)))){
          tc_V2G_SECC_Sequence_Timer.stop;
          setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                          "be executed.");
        }
          log("The PowerDeliveryReq () message content is not conform but " &
              "graceful message handling is enabled.");
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        m_CMN_AC_ChargingStatusReq_001))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(pass,"ChargingStatusReq message was correct. " &
                            "Waiting for PowerDeliveryReq message");

            v_loopCounter := v_loopCounter + 1;
            log("-------------------------------------------------------------");
            log("Number of received ChargingStatusReq messages: " &
                toString<integer>(v_loopCounter));
            log("-------------------------------------------------------------");


            if(v_loopCounter == PICS_CMN_CMN_LoopCounter) {
                setverdict(inconc, "No Renegotiation was triggered by the SUT in the " &
                                   "configured number of charging loop sequences" &
                                   "(PICS_CMN_CMN_LoopCounter).");
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            if(getverdict == pass) {
                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                if(PICS_CMN_CMN_IdentificationMode == pnC) {
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_AC_ChargingStatusRes_001(
                                    oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                    vc_MeterInfo, false, vc_AC_EVSEStatus))));
                } else {
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_AC_ChargingStatusRes_001(
                                    oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                    omit, false, vc_AC_EVSEStatus))));
                }
                repeat;
            }
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_003(vc_SessionID,
                                md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_MeteringReceiptReq_001(
                                *, vc_SessionID, *, vc_MeterInfo))))
                                -> value v_requestMessage{

            tc_V2G_SECC_Sequence_Timer.stop;
            v_isMetering := true;
            var Signature authSignature := v_requestMessage.header.signature_;
            var BodyType meteringReceiptReqBodyType := v_requestMessage.body;

            f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

            if(getverdict == pass) {

                    isSignatureOk := f_verifySignature(meteringReceiptReqBodyType,
                                                       authSignature,
                                                       vc_contractSignatureCertChain.
                                                       x509Certificate, v_vct);

                if(isSignatureOk) {
                    setverdict(pass,"MeteringReceiptReq message was correct.");
                    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_EVCC_AC_MeteringReceiptRes_001(
                                                oK,vc_AC_EVSEStatus))));
                    repeat;
                }
                else {
                    setverdict(v_vct, "Invalid signature of the MeteringReceiptReq " &
                                    "message was received.");
                }
            }
        }

        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                   {chargingStatusReq := ?},
                                                   {powerDeliveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                   "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                   "ChargingStatusReq message, " &
                                                   "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}
```

I.3.2.16 EVCC functions for MeteringReceipt

```
module TestBehavior_EVCC_MeteringReceipt {

    import from Timer_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from ComponentsAndPorts all;
    import from Pics_15118 all;
    import from Templates_EVCC_MeteringReceipt all;
    import from Templates_CMN_ChargingStatus all;
    import from Templates_CMN_CurrentDemand all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_CMN_V2GMessage all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from XSDAUX all;
    import from LibFunctions_15118_2 all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from Services_Security all;
    import from DataStructure_HAL_61851 all;

    import from Pixit_15118 all;
    import from TTlibrary_Logging all;
    import from Pixit_15118_2 all;
    import from Timer_15118 all;
    import from Templates_CMN_Security all;

    function f_EVCC_AC_TB_VTB_MeteringReceipt_001(in verdicttype v_vct)
                                                  runs on EVCC_Tester
                                                  return verdicttype {

        var charstring v_message;
        var V2G_Message v_requestMessage;
        var boolean isSignatureOk := true;
        var float v_processingDummy := 5.0;


        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_AC_ChargingStatusRes_001(
                                    oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                    vc_MeterInfo, true, vc_AC_EVSEStatus))));

        alt {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_003(vc_SessionID,
                                              md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                              md_CMN_BodyType_001(
                                              md_EVCC_CMN_MeteringReceiptReq_001(
                                              *, vc_SessionID, *, vc_MeterInfo))))
                                              -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
                var Signature authSignature := v_requestMessage.header.signature_;
                var BodyType meteringReceiptReqBodyType := v_requestMessage.body;
                vc_MeterInfo := v_requestMessage.body.bodyElement.meteringReceiptReq.
                                meterInfo;

                if(PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {

                    var SAIDType v_SASchedulTupleId := v_requestMessage.body.
                                                       bodyElement.
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                    meteringReceiptReq.
                                                    sAScheduleTupleID;

                if(v_SASchedulTupleId != vc_SAScheduleTupleId) {
                        setverdict(v_vct, "Invalid SASchedulTupleId was received " &
                                          "from the SUT.");
                }
            }

            if(ispresent(vc_MeterInfo.meterReading) and (getverdict == pass)) {
                if((vc_MeterInfo.meterReading < 0 or
                   vc_MeterInfo.meterReading > cc_maximum_value_meterReading)) {
                        setverdict(v_vct, "Invalid value range for MeterReading detected.");
                }
            }

            f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                        cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

            if(getverdict == pass) {

                    isSignatureOk := f_verifySignature(meteringReceiptReqBodyType,
                                                       authSignature,
                                                       vc_contractSignatureCertChain.
                                                       x509Certificate, v_vct);

                if(isSignatureOk) {
                    if(getverdict == pass) {
                            setverdict(pass,"MeteringReceiptReq message was correct.");
                    }
                }
                else {
                    setverdict(v_vct, "Invalid signature of the MeteringReceiptReq " &
                                      "message was received.");
                }
            }
            vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
            tc_CMN_ProcessingTime_Timer.stop;
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                         {meteringReceiptReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                              "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received MeteringReceiptReq message, " &
                                              "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_MeteringReceipt_002(urn_iso_15118_2_2013_MsgDataTypes.
                                              ResponseCodeType responseCode,
                                              in HAL_61851_Listener v_HAL_61851_Listener)
                                              runs on EVCC_Tester return verdicttype  {

    var charstring v_message;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

      f_EVCC_setPwmMode(e_PosVolt12);
      f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_EVCC_AC_MeteringReceiptRes_001(
                                 responseCode, vc_AC_EVSEStatus))));

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    alt {
        [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
           isStateB := true;
           tc_EVCC_StateB_Shutdown_Timer.stop;
           if(isClosedTCP == false) {repeat;}
        }
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
           if((isStateB == false)) {repeat;}
        }
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        m_CMN_AC_ChargingStatusReq_001))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received ChargingStatusReq message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout() {
           v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
           if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
        }
        [] a_EVCC_StateB_Shutdown_Timeout() {
           v_charstring := v_charstring & "SUT has not changed to CP State B " &
                           "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
           if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
        }
    }
    if(not(isStateB and isClosedTCP) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_EVCC_AC_TB_VTB_MeteringReceipt_003(HAL_61851_PwmMode_Type pwmMode)
                                              runs on EVCC_Tester
                                              return verdicttype  {
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_AC_MeteringReceiptRes_001(
                                    oK, vc_AC_EVSEStatus))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                        "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                m_CMN_AC_ChargingStatusReq_001))) {

               setverdict(fail,"Received ChargingStatusReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_MeteringReceipt_001(in verdicttype v_vct)
                                            runs on EVCC_Tester
                                            return verdicttype {

    var charstring v_message;
    var V2G_Message v_requestMessage;
    var boolean isSignatureOk := true;
    var float v_processingDummy := 5.0;

    log("-------------------------------------------------------------");
    log("EVSE maximum charging limits:");
    log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVSEMaximumCurrentLimit)));
    log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVSEMaximumVoltageLimit)));
    log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVSEMaximumPowerLimit)));
    log("-------------------------------------------------------------");
    if(not tc_V2G_SECC_Msg_Timer.running) {
       setverdict(inconc,"Msg performance time could not be fulfilled. " &
                         "The internal processing time should be optimized.");
      } else {tc_V2G_SECC_Msg_Timer.stop;}
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                                vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                                false, false, false, vc_EVSEMaximumVoltageLimit,
                                vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                                vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, true))));

    alt {
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_003(vc_SessionID,
                                            md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                            md_CMN_BodyType_001(
                                            md_EVCC_CMN_MeteringReceiptReq_001(
                                            *, vc_SessionID, *, vc_MeterInfo))))
                                            -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
            var Signature authSignature := v_requestMessage.header.signature_;
            var BodyType meteringReceiptReqBodyType := v_requestMessage.body;

            vc_MeterInfo := v_requestMessage.body.bodyElement.meteringReceiptReq.
                            meterInfo;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if(PIXIT_EVCC_CMN_SAScheduleTupleIDMR == sAScheduleTupleID) {

                var SAIDType v_SASchedulTupleId := v_requestMessage.body.
                                                   bodyElement.
                                                   meteringReceiptReq.
                                                   sAScheduleTupleID;

                if(v_SASchedulTupleId != vc_SAScheduleTupleId) {
                        setverdict(v_vct, "Invalid SASchedulTupleId was received " &
                                          "from the SUT.");
                }
            }

            if(ispresent(vc_MeterInfo.meterReading) and (getverdict == pass)) {
                if((vc_MeterInfo.meterReading < 0 or
                   vc_MeterInfo.meterReading > cc_maximum_value_meterReading)) {
                        setverdict(v_vct, "Invalid value range for MeterReading detected.");
                }
            }

            f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                        cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

            if(getverdict == pass) {

                    isSignatureOk := f_verifySignature(meteringReceiptReqBodyType,
                                                       authSignature,
                                                       vc_contractSignatureCertChain.
                                                       x509Certificate, v_vct);

                if(isSignatureOk) {
                    if(getverdict == pass) {
                        setverdict(pass,"MeteringReceiptReq message was correct.");
                    }
                }
                else {
                    setverdict(v_vct, "Invalid signature of the MeteringReceiptReq " &
                                      "message was received.");
                }
            }
            vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
            tc_CMN_ProcessingTime_Timer.stop;
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                       "was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                        {meteringReceiptReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received MeteringReceiptReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_MeteringReceipt_002(urn_iso_15118_2_2013_MsgDataTypes.
                                              ResponseCodeType responseCode,
                                              in HAL_61851_Listener v_HAL_61851_Listener)
                                              runs on EVCC_Tester return verdicttype  {

    var charstring v_message;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

      f_EVCC_setPwmMode(e_PosVolt12);
      f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_DC_MeteringReceiptRes_001(
                                responseCode, vc_DC_EVSEStatus))));
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
               isStateB := true;
               tc_EVCC_StateB_Shutdown_Timer.stop;
               if(isClosedTCP == false) {repeat;}
            }
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if((isStateB == false)) {repeat;}
            }
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_DC_CurrentDemandReq_001(
                                                 ?,?,*,*,*,*,?,*,*,?)))) {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(fail,"Received CurrentDemandReq message was not expected.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_MeteringReceipt_003(HAL_61851_PwmMode_Type pwmMode)
                                                 runs on EVCC_Tester
                                                 return verdicttype  {
        var boolean v_tcpTerminated := false;
        timer t_wait;

            f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
            f_EVCC_setPwmMode(pwmMode);
            t_wait.start(par_CMN_waitForNextHAL);
            alt {
               [] t_wait.timeout;
               [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                             "terminated by the SUT " &
                                                             "before request message " &
                                                             "was sent by the Test System. " &
                                                             "The send operation will be " &
                                                             "canceled.") {
                      v_tcpTerminated := true;
               }
            }
        if(not v_tcpTerminated) {
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_DC_MeteringReceiptRes_001(
                                        oK, vc_DC_EVSEStatus))));
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_DC_CurrentDemandReq_001(
                                                 ?,?,*,*,*,*,?,*,*,?)))) {

                   setverdict(fail,"Received CurrentDemandReq message was not expected. " &
                                   "CP State E/F should be detected before.");
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_EVCC_CMN_TB_VTB_MeteringReceipt_001(in HAL_61851_Listener v_HAL_61851_Listener)
                                                 runs on EVCC_Tester return verdicttype  {

        timer t_wait;
        var float v_timeShift := 1.0;
        var boolean isStateB := false;
        var boolean isClosedTCP := false;
        var charstring v_charstring := "";

        // Wait until SUT message timeout
        log("Proccessing time: " & toString<float>(vc_processingTime) & "s.");
        t_wait.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq -
                     v_timeShift - vc_processingTime);
        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));

        alt {
            [] t_wait.timeout;
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                          "terminated by the SUT " &
                                                          "before timeout has expired.");
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
                setverdict(fail, "No V2G message was expected.");
            }
        }

        if(getverdict == pass) {

              f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

            sleep(v_timeShift - par_CMN_Transmission_Delay);

            tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
                   isStateB := true;
                   tc_EVCC_StateB_Shutdown_Timer.stop;
                   if(isClosedTCP == false) {repeat;}
                }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if((isStateB == false)) {repeat;}
            }
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    }
    return getverdict;
}

function f_EVCC_CMN_TB_VTB_MeteringReceipt_002(in HAL_61851_Listener v_HAL_61851_Listener)
                                                 runs on EVCC_Tester return verdicttype {

    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

        f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
           isStateB := true;
           tc_EVCC_StateB_Shutdown_Timer.stop;
           if(isClosedTCP == false) {repeat;}
        }
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
           if((isStateB == false)) {repeat;}
        }
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                   "but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout() {
           v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
           if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }
}
```

I.3.2.17 EVCC functions for CableCheck

```
module TestBehavior_EVCC_CableCheck {

    import from Services_HAL_61851_1 all;
    import from ComponentsAndPorts all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Timer_15118_2 all;
    import from DataStructure_HAL_61851 {type IEC_61851_States;}
    import from Templates_EVCC_ChargeParameterDiscovery all;
    import from Templates_CMN_CableCheck all;
    import from Templates_CMN_PreCharge all;
    import from Pics_15118 all;
    import from Pics_15118_2 all;
    import from Templates_CMN_V2GMessage all;
    import from TTlibrary_Logging all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from DataStructure_HAL_61851 all;
    import from urn_iso_15118_2_2013_MsgHeader all;
    import from http_www_w3_org_2000_09_xmldsig all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from Services_Security all;
    import from urn_iso_15118_2_2013_X509Def all;
    import from Templates_CMN_Security all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118 all;

    import from Pixit_15118 all;
    import from DataStructure_V2G all;
    import from DataStructure_Security all;

    function f_EVCC_DC_TB_VTB_CableCheck_001(in HAL_61851_Listener v_HAL_61851_Listener,
                                             in verdicttype v_vct)
                                             runs on EVCC_Tester return verdicttype {

        var V2G_Message v_requestMessage;
        var charstring v_message;
        var SASchedules_group v_sASchedules;
        var boolean v_stateNotDetected := true;
        var MessageHeaderType v_messageHeader;
        var PhysicalValueType v_EVSEMaximumCurrentLimit := par_EVSEMaximumCurrentLimit;
        var PhysicalValueType v_EVSEMaximumPowerLimit := par_EVSEMaximumPowerLimit;

            f_EVCC_changeValidStateCondition(e_StateB,vc_state);

        if(PICS_CMN_CMN_IdentificationMode == pnC and vc_useSalesTariffs) {

            var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
            f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                                vc_myChargeParameterBodyType,
                                                vc_salesTariffIdDigestValueMap));
            var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                             vc_privateKey);
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);

        } else {
            v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
        }
        v_sASchedules.sAScheduleList := vc_SASchedule_List;
        log("-------------------------------------------------------------");
        log("EVSE maximum charging limits:");
        log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            v_EVSEMaximumCurrentLimit)));
        log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumVoltageLimit)));
        log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            v_EVSEMaximumPowerLimit)));
        log("EVSEMinimumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMinimumCurrentLimit)));
        log("EVSEMinimumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMinimumVoltageLimit)));
        log("-------------------------------------------------------------");
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        v_messageHeader, md_CMN_BodyType_001(
                                        md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                        oK, finished, v_sASchedules,
                                        md_EVCC_DC_EVSEChargeParameter_002(
                                        vc_DC_EVSEStatus,v_EVSEMaximumCurrentLimit,
                                        v_EVSEMaximumPowerLimit,
                                        vc_EVSEMaximumVoltageLimit,
                                        vc_EVSEMinimumCurrentLimit,
                                        vc_EVSEMinimumVoltageLimit,
                                        vc_EVSECurrentRegulationTolerance,
                                        vc_EVSEPeakCurrentRipple,
                                        vc_EVSEEnergyToBeDelivered)))));

        vc_DC_EVSEStatus.eVSEIsolationStatus := invalid;
        vc_DC_EVSEStatus.eVSEStatusCode := eVSE_IsolationMonitoringActive;
        alt {
            [] a_EVCC_StateCorD(
                                          pt_HAL_61851_Internal_Port, vc_state){
               v_stateNotDetected := false;
               repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(md_CMN_DC_CableCheckReq_001(
                                            md_CMN_DC_EVStatus_001(?,?,?)))))
                                            -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;

                if(v_stateNotDetected){
                    f_EVCC_StateChange_B2CorD(v_HAL_61851_Listener,
                                          par_EVCC_StateB_Shutdown_Timeout, v_vct);
                    v_stateNotDetected := false;
                }

                setverdict(pass, "CableCheckReq message was correct.");
            }
            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                           "misleadingly terminated " &
                                                           "by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                         {cableCheckReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received CableCheckReq message, " &
                                                        "but with invalid data content.");
            [] a_EVCC_Timeout(v_vct);
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_CableCheck_002(urn_iso_15118_2_2013_MsgDataTypes.
                                         ResponseCodeType responseCode,
                                         in HAL_61851_Listener v_HAL_61851_Listener)
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        runs on EVCC_Tester return verdicttype  {

    var charstring v_message;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

      f_EVCC_setPwmMode(e_PosVolt12);
      f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    vc_DC_EVSEStatus.eVSEIsolationStatus := valid;

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(md_CMN_DC_CableCheckRes_001(
                                 responseCode,finished,vc_DC_EVSEStatus))));

    alt {
        [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
           isStateB := true;
           tc_EVCC_StateB_Shutdown_Timer.stop;
           if(isClosedTCP == false) {repeat;}
        }
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
           if((isStateB == false)) {repeat;}
        }
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_PreChargeReq_001(?,?,?)))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received PreChargeReq message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                   "but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout() {
           v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
           if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
        }
        [] a_EVCC_StateB_Shutdown_Timeout() {
           v_charstring := v_charstring & "SUT has not changed to CP State B " &
                           "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
           if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
        }
    }
    if(not(isStateB and isClosedTCP) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_CableCheck_003(in HAL_61851_Listener v_HAL_61851_Listener)
                                        runs on EVCC_Tester return verdicttype  {

    timer t_wait;
    var float v_timeShift := 1.0;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

    // Wait until SUT message timeout
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    t_wait.start(par_V2G_EVCC_Msg_Timeout_CableCheckReq -
                 v_timeShift);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

          f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
               isStateB := true;
               tc_EVCC_StateB_Shutdown_Timer.stop;
               if(isClosedTCP == false) {repeat;}
            }
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if((isStateB == false)) {repeat;}
            }
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_CableCheck_004(in HAL_61851_Listener v_HAL_61851_Listener)
                                         runs on EVCC_Tester return verdicttype {

        var V2G_Message v_requestMessage;
        var charstring v_message;
        var float v_timeShift := 1.0;
        var boolean isStateB := false;
        var boolean isClosedTCP := false;
        var charstring v_charstring := "";

        vc_DC_EVSEStatus.eVSEStatusCode := eVSE_IsolationMonitoringActive;
        tc_V2G_SECC_CableCheck_Timer.start(par_V2G_EVCC_CableCheck_Timeout -
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    v_timeShift);
tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                            md_CMN_MessageHeaderType_001(vc_SessionID),
                            md_CMN_BodyType_001(md_CMN_DC_CableCheckRes_001(
                            oK, ongoing, vc_DC_EVSEStatus))));
alt {
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_DC_CableCheckReq_001(
                                      md_CMN_DC_EVStatus_001(?,?,?)))))
                                      -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            if(v_requestMessage.body.bodyElement.cableCheckReq.
               dC_EVStatus.eVReady == false) {
                logWarn("The field 'evReady' should be set to 'true'.");
            }
            setverdict(pass, "CableCheckReq message was correct.");
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_CableCheckRes_001(
                                    oK, ongoing ,vc_DC_EVSEStatus))));
            repeat;

    }
    [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                  "terminated by the SUT before " &
                                                  "timeout has expired.");
    [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {cableCheckReq := ?});
    [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                "but cannot be decoded.");
    [] a_EVCC_Unexpected_Message_Content(fail, "Received CableCheckReq message, " &
                                               "but with invalid data content.");
    [] a_EVCC_Timeout(fail);
    [] tc_V2G_SECC_CableCheck_Timer.timeout {

          f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);
          deactivate(vc_Default_IEC_61851_ListenerBehavior);

        tc_V2G_SECC_CableCheck_Timer.start(v_timeShift - par_CMN_Transmission_Delay);

        if(getverdict == pass) {

            alt {
                [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
                   isStateB := true;
                   tc_EVCC_StateB_Shutdown_Timer.stop;
                   if(isClosedTCP == false) {repeat;}
                }
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection " &
                                                              "was terminated " &
                                                              "by the SUT."){
                    setverdict(pass, "The SUT has stopped the " &
                                     "V2G Communication Session when " &
                                     "the V2G_EVCC_CableCheck_Timer " &
                                     "has expired.");
                    isClosedTCP := true;
                    tc_CMN_TCP_Connection_Termination_Timer.stop;
                    if((isStateB == false)) {repeat;}
                }
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(
                                                  vc_SessionID),
                                                  md_CMN_BodyType_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                md_CMN_DC_CableCheckReq_001(
                                                md_CMN_DC_EVStatus_001(
                                                ?,?,?))))) {

                        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(
                                                 vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_DC_CableCheckRes_001(
                                                 oK, ongoing, vc_DC_EVSEStatus))));
                        repeat;
                    }
                    [] tc_V2G_SECC_CableCheck_Timer.timeout {
                        tc_EVCC_StateB_Shutdown_Timer.start(
                        par_EVCC_StateB_Shutdown_Timeout);
                        tc_CMN_TCP_Connection_Termination_Timer.start(
                        par_CMN_TCP_Connection_Termination_Timeout);
                        repeat;
                    }
                    [] a_EVCC_Unexpected_Message_Content(fail, "Received any " &
                                                              "message, but TCP " &
                                                              "connection " &
                                                              "was >>NOT<< " &
                                                              "terminated " &
                                                              "by the SUT");
                    [] a_EVCC_TCP_Connection_Termination_Timeout() {
                       v_charstring := v_charstring & "SUT has not terminated " &
                                          "the TCP connection within " &
                                          "'par_CMN_TCP_Connection_Termination_Timeout'. ";
                       if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
                    }
                    [] a_EVCC_StateB_Shutdown_Timeout() {
                       v_charstring := v_charstring & "SUT has not changed " &
                                          "to CP State B within " &
                                          "'par_EVCC_StateB_Shutdown_Timeout'. ";
                       if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
                    }
                 }
              }
           }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

function f_EVCC_DC_TB_VTB_CableCheck_005(in HAL_61851_Listener v_HAL_61851_Listener)
                                        runs on EVCC_Tester return verdicttype {

    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

        f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
           isStateB := true;
           tc_EVCC_StateB_Shutdown_Timer.stop;
           if(isClosedTCP == false) {repeat;}
        }
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if((isStateB == false)) {repeat;}
        }
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                   "but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout() {
           v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
           if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
        }
        [] a_EVCC_StateB_Shutdown_Timeout() {
           v_charstring := v_charstring & "SUT has not changed to CP State B " &
                           "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
           if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
        }
    }
    if(not(isStateB and isClosedTCP) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_CableCheck_006(HAL_61851_PwmMode_Type pwmMode)
                                         runs on EVCC_Tester
                                         return verdicttype  {
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        vc_DC_EVSEStatus.eVSEStatusCode := eVSE_Ready;
        vc_DC_EVSEStatus.eVSEIsolationStatus := valid;

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                   md_CMN_BodyType_001(
                                   md_CMN_DC_CableCheckRes_001(
                                   oK,finished, vc_DC_EVSEStatus))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_CMN_DC_PreChargeReq_001(
                                         md_CMN_DC_EVStatus_001(?,?,?),?,?)))) {

               setverdict(fail,"Received PreChargeReq message was not expected. " &
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                "CP State E/F should be detected before.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_CableCheck_007(in HAL_61851_Listener v_HAL_61851_Listener,
                                         in verdicttype v_vct)
                                         runs on EVCC_Tester return verdicttype {

    var V2G_Message v_requestMessage;
    var charstring v_message;
    var SASchedules_group v_sASchedules;
    timer t1 := 0.1;
    timer t_cableCheck;
    timer t_keepVoltage := 3.0;
    var boolean v_isolationCheck := false;
    var boolean v_cableCheckFinished := false;
    var boolean v_stateNotDetected := true;
    var integer v_cnt := 0;
    var float v_timeShift := 1.0;
    var MessageHeaderType v_messageHeader;
    var charstring v_id;
    var PhysicalValueType v_EVSEMaximumCurrentLimit := par_EVSEMaximumCurrentLimit;
    var PhysicalValueType v_EVSEMaximumPowerLimit := par_EVSEMaximumPowerLimit;

        f_EVCC_changeValidStateCondition(e_StateB,vc_state);


    if(PICS_CMN_CMN_IdentificationMode == pnC and vc_useSalesTariffs) {

        var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
        f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                              vc_myChargeParameterBodyType,
                                              vc_salesTariffIdDigestValueMap));
        var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                         vc_privateKey);

        v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);

    } else {
        v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
    }
    v_sASchedules.sAScheduleList := vc_SASchedule_List;
    log("-------------------------------------------------------------");
    log("EVSE maximum charging limits:");
    log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        v_EVSEMaximumCurrentLimit)));
    log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVSEMaximumVoltageLimit)));
    log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        v_EVSEMaximumPowerLimit)));
    log("EVSEMinimumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVSEMinimumCurrentLimit)));
    log("EVSEMinimumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVSEMinimumVoltageLimit)));
    log("-------------------------------------------------------------");
    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                      v_messageHeader, md_CMN_BodyType_001(
                                      md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                      oK, finished, v_sASchedules,
                                      md_EVCC_DC_EVSEChargeParameter_002(
                                      vc_DC_EVSEStatus,v_EVSEMaximumCurrentLimit,
                                      v_EVSEMaximumPowerLimit,
                                      vc_EVSEMaximumVoltageLimit,
                                      vc_EVSEMinimumCurrentLimit,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    vc_EVSEMinimumVoltageLimit,
                                    vc_EVSECurrentRegulationTolerance,
                                    vc_EVSEPeakCurrentRipple,
                                    vc_EVSEEnergyToBeDelivered)))));

vc_DC_EVSEStatus.eVSEIsolationStatus := invalid;
vc_DC_EVSEStatus.eVSEStatusCode := eVSE_IsolationMonitoringActive;
alt {
    [] a_EVCC_StateCorD(
                                        pt_HAL_61851_Internal_Port, vc_state){
       v_stateNotDetected := false;
       repeat;
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(md_CMN_DC_CableCheckReq_001(
                                          md_CMN_DC_EVStatus_001(?,?,?)))))
                                          -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        v_cnt := v_cnt + 1;

        if(v_stateNotDetected){
            f_EVCC_StateChange_B2CorD(v_HAL_61851_Listener,
                                        par_EVCC_StateB_Shutdown_Timeout, v_vct);
            v_stateNotDetected := false;
        }

        if(v_cnt == 1) {
            tc_V2G_SECC_CableCheck_Timer.start(par_V2G_EVCC_CableCheck_Timeout -
                                                v_timeShift);
            t1.start;
        }


        if(v_requestMessage.body.bodyElement.cableCheckReq.
           dC_EVStatus.eVReady == false) {
            logWarn("The field 'evReady' should be set to 'true'.");
        }


        setverdict(pass, "CableCheckReq message was correct.");

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_CableCheckRes_001(
                                        oK, ongoing, vc_DC_EVSEStatus))));
        if(getverdict == pass) {
            repeat;
        }
    }
    [] t_cableCheck.timeout {
        v_cableCheckFinished := true;
        t1.start;
        repeat;
    }
    [] t1.timeout {
            if((not t_cableCheck.running) and (not v_cableCheckFinished)) {
                t_cableCheck.start(par_SECC_timeCableCheckVoltage);
            }
        if(not v_cableCheckFinished) {
            t1.start;
            repeat;

        } else {
            alt {
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(md_CMN_DC_CableCheckReq_001(
                                            md_CMN_DC_EVStatus_001(?,?,?)))))
                                            -> value v_requestMessage {

                    if(v_requestMessage.body.bodyElement.cableCheckReq.
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                dC_EVStatus.eVReady == false) {
                                 logWarn("The field 'evReady' should be set to 'true'.");
                            }
                            tc_V2G_SECC_Sequence_Timer.stop;
                            setverdict(pass, "CableCheckReq message was correct.");
                        }

                        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                                       "misleadingly terminated " &
                                                                       "by the SUT.");
                        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                                        {cableCheckReq := ?});
                        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message " &
                                                                     "was received, " &
                                                                     "but cannot be decoded.");
                        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received CableCheckReq " &
                                                                    "message, but with invalid " &
                                                                    "data content.");
                        [] a_EVCC_Timeout(v_vct);
                    }
                    vc_DC_EVSEStatus.eVSEIsolationStatus := valid;
                    vc_DC_EVSEStatus.eVSEStatusCode := eVSE_Ready;
                }
            }
            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                           "misleadingly terminated " &
                                                           "by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {cableCheckReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received CableCheckReq message, " &
                                                        "but with invalid data content.");
            [] a_EVCC_Timeout(v_vct);
        }
        return getverdict;
    }
}
```

I.3.2.18 EVCC functions for PreCharge

```
module TestBehavior_EVCC_PreCharge {

     import from Pics_15118_2 all;
     import from ComponentsAndPorts all;
     import from TestBehavior_EVCC_CommonBehavior all;
     import from TestBehavior_EVCC_PowerDelivery all;
     import from Pics_15118 all;
     import from Timer_15118_2 all;
     import from Services_HAL_61851_1 all;
     import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
     import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
     import from Templates_CMN_V2GMessage all;
     import from TestBehavior_EVCC_CommonBehavior all;
     import from Templates_CMN_CableCheck all;
     import from Templates_CMN_PreCharge all;
     import from Templates_EVCC_PaymentServiceSelection all;
     import from Templates_CMN_PowerDelivery all;
     import from TTlibrary_Logging all;
     import from Templates_CMN_V2GTcpTlsPortControl all;
     import from LibFunctions_15118_2 all;
     import from DataStructure_HAL_61851 all;
     import from Pixit_15118 all;
     import from Timer_15118 all;

     function f_EVCC_DC_TB_VTB_PreCharge_001(in verdicttype v_vct)
                                             runs on EVCC_Tester
                                             return verdicttype {

         var charstring v_message;
         var V2G_Message v_requestMessage;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
     var integer v_cnt := 0;
     tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

     pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_CableCheckRes_001(
                                oK,finished, vc_DC_EVSEStatus))));
     alt{
     [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_DC_PreChargeReq_001(
                                      md_CMN_DC_EVStatus_001(?,?,?),?,?))))
                                      -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        v_cnt := v_cnt + 1;


        vc_EVTargetCurrent := v_requestMessage.body.bodyElement.
                              preChargeReq.eVTargetCurrent;
        f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                     cc_maximum_value_physicalValue_case1,
                                     v_vct);
        vc_EVSEPresentCurrent := vc_EVTargetCurrent;
        vc_EVTargetVoltage := v_requestMessage.body.bodyElement.
                              preChargeReq.eVTargetVoltage;
        vc_EVSEPresentVoltage := vc_EVTargetVoltage;
        f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                     cc_maximum_value_physicalValue_case2,
                                     v_vct);

        if(v_requestMessage.body.bodyElement.preChargeReq.dC_EVStatus.
           eVReady == false) {
            logWarn("The field 'evReady' should be set to 'true'.");
        }
        if(getverdict == pass) {
            setverdict(pass, "PreChargeReq message was correct.");
        }
      }
     [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                    "misleadingly terminated " &
                                                    "by the SUT.");
     [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                     {preChargeReq := ?});
     [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                  "but cannot be decoded.");
     [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PreChargeReq message, " &
                                                  "but with invalid data content.");
     [] a_EVCC_Timeout(v_vct);
    }
    return getverdict
}

function f_EVCC_DC_TB_VTB_PreCharge_002(urn_iso_15118_2_2013_MsgDataTypes.
                                        ResponseCodeType responseCode,
                                        in HAL_61851_Listener v_HAL_61851_Listener)
                                        runs on EVCC_Tester return verdicttype  {

    var charstring v_message;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

      f_EVCC_setPwmMode(e_PosVolt12);
      f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_PreChargeRes_001(
                                responseCode, vc_DC_EVSEStatus,
                                vc_EVSEPresentVoltage))));
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
               isStateB := true;
               tc_EVCC_StateB_Shutdown_Timer.stop;
               if(isClosedTCP == false) {repeat;}
            }
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if((isStateB == false)) {repeat;}
            }
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_DC_PreChargeReq_001(?,?,?)))) {
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(fail,"Received PreChargeReq message was not expected.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_PreCharge_003(in HAL_61851_Listener v_HAL_61851_Listener)
                                          runs on EVCC_Tester return verdicttype  {

        timer t_wait;
        var float v_timeShift := 1.0;
        var boolean isStateB := false;
        var boolean isClosedTCP := false;
        var charstring v_charstring := "";

        // Wait until SUT message timeout
        t_wait.start(par_V2G_EVCC_Msg_Timeout_PreChargeReq -
                     v_timeShift);
        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));

        alt {
            [] t_wait.timeout;
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                          "terminated by the SUT " &
                                                          "before timeout has expired.");
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
                setverdict(fail, "No V2G message was expected.");
            }
        }

        if(getverdict == pass) {

              f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
            sleep(v_timeShift - par_CMN_Transmission_Delay);

            tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
                   isStateB := true;
                   tc_EVCC_StateB_Shutdown_Timer.stop;
                   if(isClosedTCP == false) {repeat;}
                }
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.") {
                   isClosedTCP := true;
                   tc_CMN_TCP_Connection_Termination_Timer.stop;
                   if((isStateB == false)) {repeat;}
                }
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout() {
                   v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
                   if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
                }
                [] a_EVCC_StateB_Shutdown_Timeout() {
                   v_charstring := v_charstring & "SUT has not changed to CP State B " &
                                   "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
                   if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
                }
            }
            if(not(isStateB and isClosedTCP) and v_charstring != "") {
               setverdict(fail,v_charstring);
            }
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_PreCharge_004(in HAL_61851_Listener v_HAL_61851_Listener)
                                          runs on EVCC_Tester return verdicttype  {

        var V2G_Message v_requestMessage;
        var charstring v_message;
        var float v_timeShift := 1.0;
        var boolean isStateB := false;
        var boolean isClosedTCP := false;
        var charstring v_charstring := "";

        tc_V2G_EVCC_PreCharge_Timer.start(par_V2G_EVCC_PreCharge_Timeout -
                                         v_timeShift);
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

        vc_EVSEPresentVoltage := { multiplier := 0, unit := v, value_ := 0 };

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_PreChargeRes_001(
                                    oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage))));
        alt {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
                                              md_CMN_BodyType_001(
                                              md_CMN_DC_PreChargeReq_001(
                                              md_CMN_DC_EVStatus_001(?,?,?),?,?))))
                                              -> value v_requestMessage {

                    tc_V2G_SECC_Sequence_Timer.stop;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        vc_EVTargetCurrent := v_requestMessage.body.bodyElement.
                              preChargeReq.eVTargetCurrent;
        f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                         cc_maximum_value_physicalValue_case1,
                                         fail);
        vc_EVSEPresentCurrent := vc_EVTargetCurrent;
        vc_EVTargetVoltage := v_requestMessage.body.bodyElement.
                              preChargeReq.eVTargetVoltage;
        f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                         cc_maximum_value_physicalValue_case2,
                                         fail);
        if(v_requestMessage.body.bodyElement.preChargeReq.dC_EVStatus.
           eVReady == false) {
            logWarn("The field 'evReady' should be set to 'true'.");
        }
        if(getverdict == pass){
            setverdict(pass, "PreChargeReq message was correct.");
        }
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_PreChargeRes_001(
                                    oK, vc_DC_EVSEStatus,
                                    vc_EVSEPresentVoltage))));
        repeat;

    }
    [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                    "terminated by the SUT before " &
                                    "timeout has expired.");
    [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                  {preChargeReq := ?});
    [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                            "but cannot be decoded.");
    [] a_EVCC_Unexpected_Message_Content(fail, "Received PreChargeReq message, " &
                                           "but with invalid data content.");
    [] a_EVCC_Timeout(fail);
    [] tc_V2G_EVCC_PreCharge_Timer.timeout {

          f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);
          deactivate(vc_Default_IEC_61851_ListenerBehavior);

        tc_V2G_EVCC_PreCharge_Timer.start(v_timeShift - par_CMN_Transmission_Delay);

        if(getverdict == pass) {

            alt {
                [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
                   isStateB := true;
                   tc_EVCC_StateB_Shutdown_Timer.stop;
                   if(isClosedTCP == false) {repeat;}
                }
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                        "terminated by the SUT."){
                    setverdict(pass, "The SUT has stopped the " &
                                     "V2G Communication Session when " &
                                     "the V2G_EVCC_PreCharge_Timer has expired.");
                    isClosedTCP := true;
                    tc_CMN_TCP_Connection_Termination_Timer.stop;
                    if((isStateB == false)) {repeat;}
                }
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(
                                                vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_DC_PreChargeReq_001(
                                                md_CMN_DC_EVStatus_001(?,?,?),
                                                ?,?)))) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                          md_CMN_MessageHeaderType_001(
                                                          vc_SessionID),
                                                          md_CMN_BodyType_001(
                                                          md_CMN_DC_PreChargeRes_001(
                                                          oK, vc_DC_EVSEStatus,
                                                          vc_EVSEPresentVoltage))));
                        repeat;
                    }
                    [] tc_V2G_EVCC_PreCharge_Timer.timeout {
                        tc_EVCC_StateB_Shutdown_Timer.start(
                        par_EVCC_StateB_Shutdown_Timeout);
                        tc_CMN_TCP_Connection_Termination_Timer.start(
                        par_CMN_TCP_Connection_Termination_Timeout);
                        repeat;
                    }
                    [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                               "but TCP connection " &
                                                               "was >>NOT<< terminated " &
                                                               "by the SUT");
                    [] a_EVCC_TCP_Connection_Termination_Timeout() {
                       v_charstring := v_charstring & "SUT has not terminated " &
                                       "the TCP connection within " &
                                       "'par_CMN_TCP_Connection_Termination_Timeout'. ";
                       if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
                    }
                    [] a_EVCC_StateB_Shutdown_Timeout() {
                       v_charstring := v_charstring & "SUT has not changed " &
                                       "to CP State B within " &
                                       "'par_EVCC_StateB_Shutdown_Timeout'. ";
                       if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
                    }
                }
            }
        }
    }
    if(not(isStateB and isClosedTCP) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_PreCharge_005(in HAL_61851_Listener v_HAL_61851_Listener)
                                        runs on EVCC_Tester return verdicttype {

    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

        f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
           isStateB := true;
           tc_EVCC_StateB_Shutdown_Timer.stop;
           if(isClosedTCP == false) {repeat;}
        }
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
           if((isStateB == false)) {repeat;}
        }
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_PreCharge_006(HAL_61851_PwmMode_Type pwmMode)
                                            runs on EVCC_Tester
                                            return verdicttype  {
        var boolean v_tcpTerminated := false;
        timer t_wait;

            f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
            f_EVCC_setPwmMode(pwmMode);
            t_wait.start(par_CMN_waitForNextHAL);
            alt {
               [] t_wait.timeout;
               [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                             "terminated by the SUT " &
                                                             "before request message " &
                                                             "was sent by the Test System. " &
                                                             "The send operation will be " &
                                                             "canceled.") {
                   v_tcpTerminated := true;
               }
            }

        if(not v_tcpTerminated) {
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_PreChargeRes_001(
                                        oK, vc_DC_EVSEStatus,
                                        vc_EVSEPresentVoltage))));

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
                                              md_CMN_BodyType_001(
                                              md_CMN_DC_PreChargeReq_001(?,?,?)))) {

                   setverdict(fail,"Received PreChargeReq message was not expected. " &
                                   "CP State E/F should be detected before.");
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                          "but TCP connection " &
                                                          "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }
}
```

I.3.2.19 EVCC functions for PreChargeOrPowerDelivery

```
module TestBehavior_EVCC_PreChargeOrPowerDelivery {

     import from Services_HAL_61851_1 all;
     import from Timer_15118_2  all;
     import from ComponentsAndPorts all;
     import from Templates_CMN_CableCheck all;
     import from Templates_CMN_PreCharge all;
     import from Templates_CMN_PowerDelivery all;
     import from Templates_EVCC_PaymentServiceSelection all;
     import from Templates_CMN_V2GMessage all;
     import from Templates_CMN_CurrentDemand all;
     import from Timer_15118 all;
     import from TestBehavior_EVCC_CommonBehavior all;
     import from TestBehavior_EVCC_PowerDelivery all;
     import from Pics_15118 all;
     import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
     import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
     import from TTlibrary_Logging all;
     import from Templates_CMN_V2GTcpTlsPortControl all;
     import from LibFunctions_15118_2 all;
     import from DataStructure_HAL_61851 all;
     import from Pixit_15118 all;
     import from Pixit_15118_2 all;

     function f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_001(in verdicttype v_vct)
                                                            runs on EVCC_Tester
                                                            return verdicttype {
       var charstring v_message;
       var V2G_Message v_requestMessage;
       vc_EVSEPresentVoltage.multiplier := 0;
       vc_EVSEPresentVoltage.value_ := 0;
       var integer v_voltage_temp := 0;
       var integer v_cnt := 0;

       var integer v_voltageStep := float2int(f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage))/
                                              cc_preChargeVoltageRampUpSteps;

       tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
       tc_V2G_EVCC_PreCharge_Timer.start(par_V2G_EVCC_PreCharge_Timeout);

       pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_DC_PreChargeRes_001(
                                      oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage))));

       alt {
         [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_DC_PowerDeliveryReq_002(
                                                 start_, ?, ?,
                                                 md_CMN_DC_EVPowerDeliveryParameter(
                                                 md_CMN_DC_EVStatus_001(?,?,?),*,false)))))
                                                 -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            if(v_requestMessage.body.bodyElement.powerDeliveryReq.eVPowerDeliveryParameter.
               dC_EVPowerDeliveryParameter.dC_EVStatus.eVReady == false) {
               logWarn("The field 'evReady' should be set to 'true'.");
            }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        vc_chargingProfile := v_requestMessage.body.bodyElement.
                              powerDeliveryReq.chargingProfile;

        var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                            powerDeliveryReq.sAScheduleTupleID;


        if(PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization) {
            if(v_SAScheduleTupleId != vc_SAScheduleTupleId3) {
                    setverdict(v_vct, "Wrong SAScheduleTupleId was " &
                                      "chosen from SUT. The SalesTariff " &
                                      "with the lowest price level " &
                                      "was not considered.");
            }
        } else if(PIXIT_EVCC_CMN_ChargingProfileOptimization == none_) {
            if(v_SAScheduleTupleId != vc_SAScheduleTupleId) {
                    setverdict(v_vct, "Wrong SAScheduleTupleId was " &
                                      "chosen from SUT. The SUT did not choose " &
                                      "the default SAScheduleTuple.");
            }
        } else {
            if((v_SAScheduleTupleId == vc_SAScheduleTupleId) or
               (v_SAScheduleTupleId == vc_SAScheduleTupleId2) or
               (v_SAScheduleTupleId == vc_SAScheduleTupleId3)) {
                setverdict(pass, "Valid SAScheduleTupleId was " &
                                 "chosen from SUT.");
            } else {
                    setverdict(v_vct, "Invalid SAScheduleTupleId was " &
                                      "chosen from SUT.");
            }
        }
        vc_SAScheduleTupleId := v_SAScheduleTupleId;

        if(getverdict == pass) {
            f_checkValidMaxPowerEntries(vc_chargingProfile, vc_SASchedule_List,
                                        vc_SAScheduleTupleId, v_vct);
        }
        if(getverdict == pass) {
            setverdict(pass, "PowerDeliveryReq message was correct.");
        }
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        mdw_CMN_DC_PowerDeliveryReq_003(
                                        stop_, vc_SAScheduleTupleId, *))))
                                        -> value v_requestMessage {

         tc_V2G_SECC_Sequence_Timer.stop;
         if(PIXIT_EVCC_CMN_Pause == pause or
               PICS_EVCC_CMN_PmaxSchedulewithZeroPow == sleepWithoutCharge) {
             vc_sleepModeWoCharging := true;
             setverdict(pass,"PowerDeliveryReq message with ChargeProgressType " &
                             "'Stop_' was received. The SUT wants to initiate a " &
                             "sleep phase directly.");
         } else {
             setverdict(inconc,"PowerDeliveryReq message with ChargeProgressType " &
                               "'Stop_' was received. The SUT wants to initiate a " &
                               "sleep phase directly.");
         }
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_PreChargeReq_001(
                                        md_CMN_DC_EVStatus_001(?,?,?), ?, ?))))
                                        -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        v_cnt := v_cnt + 1;
        vc_EVTargetVoltage := v_requestMessage.body.bodyElement.
                              preChargeReq.eVTargetVoltage;
        if(v_cnt <= cc_preChargeVoltageRampUpSteps) {
          v_voltage_temp := v_voltage_temp + v_voltageStep;
          vc_EVSEPresentVoltage.value_ := v_voltage_temp;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
          }
          if(v_cnt > cc_preChargeVoltageRampUpSteps) {
            vc_EVSEPresentVoltage.value_ := float2int(f_CMN_PhysicalValue_GetValue(
                                              vc_EVTargetVoltage));
          }
          if(v_requestMessage.body.bodyElement.preChargeReq.dC_EVStatus.
             eVReady == false) {
              logWarn("The field 'evReady' should be set to 'true'.");
          }
          vc_EVTargetCurrent := v_requestMessage.body.bodyElement.preChargeReq.
                                  eVTargetCurrent;
          f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                            cc_maximum_value_physicalValue_case1,
                                            v_vct);
          vc_EVSEPresentCurrent := vc_EVTargetCurrent;
          f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                            cc_maximum_value_physicalValue_case2,
                                            v_vct);
          if(getverdict == pass) {
             setverdict(pass,"PreChargeReq message was correct. " &
                             "Waiting for PowerDeliveryReq message");
          }
          tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
          // repeat sending pre-charge response message
          pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_PreChargeRes_001(
                                          oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage))));
          repeat;
       }
       [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
       [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                  {preChargeReq := ?}, {powerDeliveryReq := ?});
       [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                    "but cannot be decoded.");
       [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                   "PreChargeReq message, but with " &
                                                   "invalid data content.");
       [] a_EVCC_Timeout(v_vct);
       [] tc_V2G_EVCC_PreCharge_Timer.timeout {
          tc_V2G_SECC_Sequence_Timer.stop;
          setverdict(v_vct, "PreChargeTimer should be triggered on SUT side. " &
                            "PreCharge process was not successful.");
       }
    }

    return getverdict;
  }

function f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_002(urn_iso_15118_2_2013_MsgDataTypes.
                                                        ResponseCodeType responseCode,
                                                        in HAL_61851_Listener
                                                        v_HAL_61851_Listener)
                                                        runs on EVCC_Tester
                                                        return verdicttype {

    var charstring v_message;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

      f_EVCC_setPwmMode(e_PosVolt12);
      f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_PowerDeliveryRes_001(
                                responseCode,
                                md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
               isStateB := true;
               tc_EVCC_StateB_Shutdown_Timer.stop;
               if(isClosedTCP == false) {repeat;}
            }
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if((isStateB == false)) {repeat;}
            }
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_CMN_DC_CurrentDemandReq_001(
                                                 ?,?,*,*,*,*,?,*,*,?)))) {

                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(fail,"Received CurrentDemandReq message was not expected.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_003(in HAL_61851_Listener
                                                            v_HAL_61851_Listener)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

        timer t_wait;
        var float v_timeShift := 1.0;
        var boolean isStateB := false;
        var boolean isClosedTCP := false;
        var charstring v_charstring := "";

        // Wait until SUT message timeout
        t_wait.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq -
                     v_timeShift);
        pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                    e_getPortStatus));

        alt {
            [] t_wait.timeout;
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                          "terminated by the SUT " &
                                                          "before timeout has expired.");
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
                setverdict(fail, "No V2G message was expected.");
            }
        }

        if(getverdict == pass) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
               isStateB := true;
               tc_EVCC_StateB_Shutdown_Timer.stop;
               if(isClosedTCP == false) {repeat;}
            }
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if((isStateB == false)) {repeat;}
            }
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_004(in HAL_61851_Listener
                                                       v_HAL_61851_Listener)
                                                       runs on EVCC_Tester
                                                       return verdicttype {

    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

        f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
           isStateB := true;
           tc_EVCC_StateB_Shutdown_Timer.stop;
           if(isClosedTCP == false) {repeat;}
        }
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            isClosedTCP := true;
            tc_CMN_TCP_Connection_Termination_Timer.stop;
            if((isStateB == false)) {repeat;}
        }
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout() {
           v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
           if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
        }
        [] a_EVCC_StateB_Shutdown_Timeout() {
           v_charstring := v_charstring & "SUT has not changed to CP State B " &
                           "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
           if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
        }
    }
    if(not(isStateB and isClosedTCP) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_005(HAL_61851_PwmMode_Type pwmMode)
                                                       runs on EVCC_Tester
                                                       return verdicttype  {
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_CMN_PowerDeliveryRes_001(
                                  oK, md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_CurrentDemandReq_001(
                                        ?,?,*,*,*,*,?,*,*,?)))) {

               setverdict(fail,"Received CurrentDemandReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
         }
      }
      return getverdict;
   }

   function f_EVCC_DC_TB_VTB_PreChargeOrPowerDelivery_006() runs on EVCC_Tester
                                                            return verdicttype {
     var charstring v_message;
     var V2G_Message v_requestMessage;
     vc_EVSEPresentVoltage.multiplier := 0;
     vc_EVSEPresentVoltage.value_ := 0;
     var integer v_voltage_temp := 0;
     var integer v_cnt := 0;

     var integer v_voltageStep := float2int(f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage))/
                                          cc_preChargeVoltageRampUpSteps;


     tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
     tc_V2G_EVCC_PreCharge_Timer.start(par_V2G_EVCC_PreCharge_Timeout);

     pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_PreChargeRes_001(
                                oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage))));

     alt {
       [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_PowerDeliveryReq_002(
                                          start_, ?, ?,
                                          md_CMN_DC_EVPowerDeliveryParameter(
                                          md_CMN_DC_EVStatus_001(?,?,?),*,false)))))
                                          -> value v_requestMessage {

          tc_V2G_SECC_Sequence_Timer.stop;
          if(v_requestMessage.body.bodyElement.powerDeliveryReq.eVPowerDeliveryParameter.
             dC_EVPowerDeliveryParameter.dC_EVStatus.eVReady == false) {
             logWarn("The field 'evReady' should be set to 'true'.");
          }
          var ChargingProfileType v_chargingProfile := v_requestMessage.
                                                        body.bodyElement.
                                                       powerDeliveryReq.
                                                       chargingProfile;

          var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                              powerDeliveryReq.sAScheduleTupleID;

          if(PIXIT_EVCC_CMN_OldSAScheduleTupleID == unknown) {

             if(PIXIT_EVCC_CMN_ChargingProfileOptimization == optimization) {
                if(v_SAScheduleTupleId != vc_SAScheduleTupleIdRenegotiation3) {
                    setverdict(fail, "Wrong SAScheduleTupleId was " &
                                     "chosen from SUT. The SalesTariff " &
                                     "with the lowest price level " &
                                     "was not considered.");
                }
             } else if(PIXIT_EVCC_CMN_ChargingProfileOptimization == none_) {
                if(v_SAScheduleTupleId != vc_SAScheduleTupleIdRenegotiation) {
                    setverdict(fail, "Wrong SAScheduleTupleId was " &
                                     "chosen from SUT. The SUT did not choose " &
                                     "the default SAScheduleTuple.");
                }
             } else {
                if((v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation) or
                   (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation2) or
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation3)) {
                 setverdict(pass, "Valid SAScheduleTupleId was " &
                                  "chosen from SUT.");
            } else {
                 setverdict(fail, "Invalid SAScheduleTupleId was " &
                                  "chosen from SUT.");
            }
        }
      } else {
        if(v_chargingProfile != vc_chargingProfile) {
            setverdict(fail, "The received charging profile is not equal " &
                             "to the selected charging profile before " &
                             "renegotiation.");
        }
      }
      vc_SAScheduleTupleId := v_SAScheduleTupleId;
      vc_chargingProfile := v_chargingProfile;

      if(getverdict == pass) {
          f_checkValidMaxPowerEntries(vc_chargingProfile, vc_SASchedule_List,
                                      vc_SAScheduleTupleId, fail);
          setverdict(pass, "PowerDeliveryReq message was correct.");
      }
}
[] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_PreChargeReq_001(
                                    md_CMN_DC_EVStatus_001(?,?,?), ?, ?))))
                                    -> value v_requestMessage {

    tc_V2G_SECC_Sequence_Timer.stop;
    v_cnt := v_cnt + 1;
    vc_EVTargetVoltage := v_requestMessage.body.bodyElement.
                          preChargeReq.eVTargetVoltage;
    if(v_cnt <= cc_preChargeVoltageRampUpSteps) {
      v_voltage_temp := v_voltage_temp + v_voltageStep;
      vc_EVSEPresentVoltage.value_ := v_voltage_temp;
    }
    if(v_cnt > cc_preChargeVoltageRampUpSteps) {
      vc_EVSEPresentVoltage.value_ := float2int(f_CMN_PhysicalValue_GetValue(
                                      vc_EVTargetVoltage));
    }
    if(v_requestMessage.body.bodyElement.preChargeReq.dC_EVStatus.
       eVReady == false) {
        logWarn("The field 'evReady' should be set to 'true'.");
    }
    vc_EVTargetCurrent := v_requestMessage.body.bodyElement.preChargeReq.
                          eVTargetCurrent;
    f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                     cc_maximum_value_physicalValue_case1,
                                     fail);
    vc_EVSEPresentCurrent := vc_EVTargetCurrent;

    f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                     cc_maximum_value_physicalValue_case2,
                                     fail);
    if(getverdict == pass) {
       setverdict(pass,"PreChargeReq message was correct. " &
                       "Waiting for PowerDeliveryReq message");
    }

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    // repeat sending pre-charge response message
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_PreChargeRes_001(
                                oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage))));
    repeat;
}
[] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                        "terminated by the SUT.");
[] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(fail, "Unexpected V2G message was received.",
                                            {preChargeReq := ?}, {powerDeliveryReq := ?});
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PowerDeliveryReq or " &
                                                   "PreChargeReq message, but with " &
                                                   "invalid data content.");
        [] a_EVCC_Timeout(fail);
        [] tc_V2G_EVCC_PreCharge_Timer.timeout {
           tc_V2G_SECC_Sequence_Timer.stop;
           setverdict(fail, "PreChargeTimer should be triggered on SUT side. " &
                            "PreCharge process was not successful.");
        }
      }

      return getverdict;
    }
}
```

I.3.2.20 EVCC functions for CurrentDemand

```
module TestBehavior_EVCC_CurrentDemand {

    import from Services_HAL_61851_1 all;
    import from Timer_15118_2 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from Templates_CMN_CurrentDemand all;
    import from Templates_CMN_CableCheck all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_EVCC_MeteringReceipt all;
    import from Templates_CMN_V2GMessage all;
    import from Pics_15118 all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from TTlibrary_Logging all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from LibFunctions_15118_2 all;
    import from DataStructure_HAL_61851 all;
    import from Timer_15118 all;
    import from Pixit_15118 all;

    function f_EVCC_DC_TB_VTB_CurrentDemand_001(in verdicttype v_vct)
                                                runs on EVCC_Tester
                                                return verdicttype {

        var V2G_Message v_requestMessage;
        var charstring v_message;
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_PowerDeliveryRes_001(
                                    oK, md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));
           alt {
          [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_DC_CurrentDemandReq_001(
                                                md_CMN_DC_EVStatus_001(?,?,?),
                                                ?,*,*,*,*,?,*,*,?))))
                                                -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                tc_V2G_SECC_Msg_Timer.start(
                par_V2G_SECC_Msg_CurrentDemand_Performance_Time);
                vc_EVTargetCurrent := v_requestMessage.body.bodyElement.currentDemandReq.
                                      eVTargetCurrent;
                f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                                 cc_maximum_value_physicalValue_case1,
                                                 v_vct);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            vc_EVSEPresentCurrent := vc_EVTargetCurrent;
            vc_EVTargetVoltage := v_requestMessage.body.bodyElement.currentDemandReq.
                                  eVTargetVoltage;
            f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                             cc_maximum_value_physicalValue_case2,
                                             v_vct);

            if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
                         eVMaximumCurrentLimit)) {
                vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                                            currentDemandReq.eVMaximumCurrentLimit;
                f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                                 vc_EVMaximumCurrentLimit, a,
                                                 cc_maximum_value_physicalValue_case1,
                                                 v_vct);
            }
            if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
                         eVMaximumVoltageLimit)) {
                vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                                            currentDemandReq.eVMaximumVoltageLimit;
                f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                                 vc_EVMaximumVoltageLimit, v,
                                                 cc_maximum_value_physicalValue_case2,
                                                 v_vct);
            }
            if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
                         eVMaximumPowerLimit)) {
                vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                          currentDemandReq.eVMaximumPowerLimit;
                f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                                 vc_EVMaximumPowerLimit, w,
                                                 cc_maximum_value_physicalValue_case4,
                                                 v_vct);
            }
            if(v_requestMessage.body.bodyElement.currentDemandReq.dC_EVStatus.
               eVReady == false) {
                logWarn("The field 'evReady' should be set to 'true'.");
            }
            log("-------------------------------------------------------------");
            log("EV maximum charging limits:");
            log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumCurrentLimit)));
            log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumVoltageLimit)));
            if(vc_isPowLimit) {
                log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                    vc_EVMaximumPowerLimit)));
            }
            log("-------------------------------------------------------------");
            if(getverdict == pass) {
                  setverdict(pass, "CurrentDemandReq message was correct.");
            }
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                  "misleadingly terminated " &
                                                  "by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                         {currentDemandReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                              "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received CurrentDemandReq message, " &
                                               "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_CurrentDemand_002(urn_iso_15118_2_2013_MsgDataTypes.
                                                ResponseCodeType responseCode,
                                                in HAL_61851_Listener v_HAL_61851_Listener)
                                                runs on EVCC_Tester return verdicttype  {

        var charstring v_message;
        var boolean isStateB := false;
        var boolean isClosedTCP := false;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    var charstring v_charstring := "";

      f_EVCC_setPwmMode(e_PosVolt12);
      f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_CurrentDemandRes_001(
                                responseCode, vc_DC_EVSEStatus,
                                vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                                false, false, false, omit, omit, omit, vc_EvseId,
                                vc_SAScheduleTupleId, omit, omit))));

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
           isStateB := true;
           tc_EVCC_StateB_Shutdown_Timer.stop;
           if(isClosedTCP == false) {repeat;}
        }
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
           if((isStateB == false)) {repeat;}
        }
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_CMN_DC_CurrentDemandReq_001(
                                         ?,?,*,*,*,*,?,*,*,?)))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received CurrentDemandReq message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout() {
           v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
           if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
        }
        [] a_EVCC_StateB_Shutdown_Timeout() {
           v_charstring := v_charstring & "SUT has not changed to CP State B " &
                           "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
           if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
        }
    }
    if(not(isStateB and isClosedTCP) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_CurrentDemand_003(in HAL_61851_Listener v_HAL_61851_Listener)
                                         runs on EVCC_Tester return verdicttype  {

    timer t_wait;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

    // Wait until SUT message timeout

      f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                e_getPortStatus));

    sleep(par_V2G_EVCC_Msg_Timeout_CurrentDemandReq - par_CMN_Transmission_Delay);

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
           isStateB := true;
           tc_EVCC_StateB_Shutdown_Timer.stop;
           if(isClosedTCP == false) {repeat;}
        }
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
           if((isStateB == false)) {repeat;}
        }
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout() {
           v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
           if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
        }
        [] a_EVCC_StateB_Shutdown_Timeout() {
           v_charstring := v_charstring & "SUT has not changed to CP State B " &
                           "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
           if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
        }
    }
    if(not(isStateB and isClosedTCP) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_CurrentDemand_004(HAL_61851_PwmMode_Type pwmMode)
                                            runs on EVCC_Tester
                                            return verdicttype  {

    var float v_time := 0.1;
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(v_time);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        if(PICS_CMN_CMN_IdentificationMode == pnC) {
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                              md_CMN_MessageHeaderType_001(vc_SessionID),
                              md_CMN_BodyType_001(
                              md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                              vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                              false, false, false, vc_EVSEMaximumVoltageLimit,
                              vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, false))));
            } else {
               pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                                vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                                false, false, false, vc_EVSEMaximumVoltageLimit,
                                vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                                vc_EvseId, vc_SAScheduleTupleId, omit, false))));
            }

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt {
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_CMN_SessionStop_Termination() {
                    tc_CMN_TCP_Connection_Termination_Timer.start(
                    par_CMN_TCP_Connection_Termination_Timeout);
                    repeat;
                }
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                                  md_CMN_BodyType_001(
                                                  md_CMN_DC_CurrentDemandReq_001(
                                                  md_CMN_DC_EVStatus_001(?,?,?),
                                                  ?,*,*,*,*,?,*,*,?)))) {
                   setverdict(fail,"Received ChargeParamterDiscoveryReq message was not expected. " &
                                   "CP State E/F should be detected before.");
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT") {
                   log("CP State E/F should be detected before.");
                }
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_CurrentDemand_005(in HAL_61851_Listener v_HAL_61851_Listener)
                                             runs on EVCC_Tester return verdicttype {

        var boolean isStateB := false;
        var boolean isClosedTCP := false;
        var charstring v_charstring := "";

            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

        tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
               isStateB := true;
               tc_EVCC_StateB_Shutdown_Timer.stop;
               if(isClosedTCP == false) {repeat;}
            }
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if((isStateB == false)) {repeat;}
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
               v_charstring := v_charstring & "SUT has not changed to CP State B " &
                               "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
               if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
         }
         if(not(isStateB and isClosedTCP) and v_charstring != "") {
            setverdict(fail,v_charstring);
         }
         return getverdict;
      }
}
```

I.3.2.21 EVCC functions for ChargingLoopDcOrPowerDelivery

```
module TestBehavior_EVCC_ChargingLoopDcOrPowerDelivery {

  import from Pics_15118 all;
  import from Services_HAL_61851_1 all;
  import from LibFunctions_15118_2 all;
  import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
  import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
  import from Templates_CMN_V2GMessage all;
  import from Templates_CMN_PreCharge all;
  import from Templates_EVCC_ChargeParameterDiscovery all;
  import from Templates_CMN_CurrentDemand all;
  import from Templates_EVCC_MeteringReceipt all;
  import from Templates_EVCC_PaymentServiceSelection all;
  import from Templates_CMN_PowerDelivery all;
  import from Templates_CMN_WeldingDetection all;
  import from Templates_CMN_SessionStop all;
  import from TestBehavior_EVCC_CommonBehavior all;
  import from Templates_CMN_PreCharge all;
  import from TestBehavior_EVCC_PowerDelivery all;
  import from Pics_15118_2 all;
  import from ComponentsAndPorts all;
  import from Timer_15118_2 all;
  import from DataStructure_HAL_61851 all;
  import from urn_iso_15118_2_2013_MsgDef all;
  import from TTlibrary_Logging all;
  import from Templates_CMN_V2GTcpTlsPortControl all;
  import from Templates_CMN_CableCheck all;
  import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
  import from Services_Security all;
  import from DataStructure_HAL_61851 all;
  import from urn_iso_15118_2_2013_MsgHeader all;
  import from http_www_w3_org_2000_09_xmldsig all;
  import from Services_Security all;
  import from urn_iso_15118_2_2013_X509Def all;
  import from Templates_CMN_Security all;

  import from Pixit_15118 all;
  import from Timer_15118 all;
  import from DataStructure_V2G all;
  import from TestBehavior_EVCC_ChargeParameterDiscovery all;

  function f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_001(in HAL_61851_Listener
                                                                 v_HAL_61851_Listener,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                                    in verdicttype v_vct)
                                                                    runs on EVCC_Tester
                                                                    return verdicttype {

    var charstring v_message;
    var V2G_Message v_requestMessage;
    var integer v_loopCounter := 1;
    var integer v_renegotiationCounter := 0;
    var boolean v_isCableCheck := false;
    var boolean v_isPreCharge := false;
    var MessageHeaderType v_messageHeader;
    var charstring v_id;
    var SASchedules_group v_sASchedules;
    var boolean v_isFinished := true;
    timer t1 := 0.1;
    timer t_cableCheck;
    timer t_keepVoltage := 3.0;
    var boolean v_isolationCheck := false;
    var boolean v_cableCheckFinished := false;
    var boolean v_stateNotDetected := true;
    var integer v_cnt := 0;
    var integer v_voltage_temp := 0;
    var integer v_cnt_PC := 0;
    timer t_stop;
    var timer t_stopProcessing;
    var float v_processingDummy := 5.0;

    var integer v_voltageStep := float2int(f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage))/
                                        cc_preChargeVoltageRampUpSteps;

    log("-------------------------------------------------------------");
    log("Number of received CurrentDemandReq messages: " & toString<integer>(v_loopCounter));
    log("-------------------------------------------------------------");
    if(not tc_V2G_SECC_Msg_Timer.running) {
        setverdict(inconc,"Msg performance time could not be fulfilled. " &
                          "The internal processing time should be optimized.");
    } else {tc_V2G_SECC_Msg_Timer.stop;}
    log("-------------------------------------------------------------");
    log("EVSE maximum charging limits:");
    log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVSEMaximumCurrentLimit)));
    log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVSEMaximumVoltageLimit)));
    log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVSEMaximumPowerLimit)));
    log("-------------------------------------------------------------");

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    if(PICS_CMN_CMN_IdentificationMode == pnC) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                          md_CMN_MessageHeaderType_001(vc_SessionID),
                          md_CMN_BodyType_001(
                          md_CMN_DC_CurrentDemandRes_001(
                          oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage,
                          vc_EVSEPresentCurrent, false, false, false,
                          vc_EVSEMaximumVoltageLimit, vc_EVSEMaximumCurrentLimit,
                          vc_EVSEMaximumPowerLimit, vc_EvseId, vc_SAScheduleTupleId,
                          vc_MeterInfo, false))));
    } else {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                          md_CMN_MessageHeaderType_001(vc_SessionID),
                          md_CMN_BodyType_001(
                          md_CMN_DC_CurrentDemandRes_001(
                          oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage,
                          vc_EVSEPresentCurrent, false, false, false,
                          vc_EVSEMaximumVoltageLimit, vc_EVSEMaximumCurrentLimit,
                          vc_EVSEMaximumPowerLimit, vc_EvseId, vc_SAScheduleTupleId,
                          omit, false))));
    }

    alt {
      [not vc_renegotiation] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                        md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        mdw_CMN_DC_PowerDeliveryReq_003(
                                        stop_, vc_SAScheduleTupleId, *))))
                                      -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
        setverdict(pass, "PowerDeliveryReq message was correct.");
        vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
        tc_CMN_ProcessingTime_Timer.stop;
    }
    [v_renegotiationCounter == 0]
    pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_PowerDeliveryReq_002(
                                  renegotiate, vc_SAScheduleTupleId, *,
                                  md_CMN_DC_EVPowerDeliveryParameter(
                                  md_CMN_DC_EVStatus_001(?,?,?),*,?))))) {

        tc_V2G_SECC_Sequence_Timer.stop;
        t_stopProcessing.stop;
        vc_renegotiation := true;
        v_renegotiationCounter := v_renegotiationCounter + 1;


        setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                        "be executed.");

            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_CMN_PowerDeliveryRes_001(oK,
                                      md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

            f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener,
                                    par_V2G_SECC_Sequence_Timeout, v_vct);
        vc_DC_EVSEStatus.eVSENotification := none_;
        repeat;
    }
    [vc_renegotiation and v_renegotiationCounter == 1]
    pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                    *,?,?)))) -> value v_requestMessage {

        v_isFinished := true;
        vc_DC_EVSEStatus := {par_EVCC_NotificationMaxDelay, none_, invalid, eVSE_Ready};
        tc_V2G_SECC_Sequence_Timer.stop;
        if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                     maxEntriesSAScheduleTuple)) {
            vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                            chargeParameterDiscoveryReq.
                                            maxEntriesSAScheduleTuple;
        }
        v_renegotiationCounter := v_renegotiationCounter + 1;
        var EVChargeParameter_group evChargeParameter := v_requestMessage.body.bodyElement.
                                                         chargeParameterDiscoveryReq.
                                                         eVChargeParameter;
        if(evChargeParameter.dC_EVChargeParameter.dC_EVStatus.eVReady == true) {
            logWarn("The field 'evReady' should be set to 'false'.");
        }
        vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                        chargeParameterDiscoveryReq.eVChargeParameter.
                        dC_EVChargeParameter.eVMaximumCurrentLimit;
        f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                         vc_EVMaximumCurrentLimit, a,
                                         cc_maximum_value_physicalValue_case1,
                                         v_vct);
        vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
        chargeParameterDiscoveryReq.eVChargeParameter.dC_EVChargeParameter.
        eVMaximumVoltageLimit;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                 vc_EVMaximumVoltageLimit, v,
                                 cc_maximum_value_physicalValue_case2,
                                 v_vct);
if(ispresent(v_requestMessage.body.bodyElement.
             chargeParameterDiscoveryReq.eVChargeParameter.dC_EVChargeParameter.
             eVMaximumPowerLimit)) {
      vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                chargeParameterDiscoveryReq.eVChargeParameter.
                                dC_EVChargeParameter.eVMaximumPowerLimit;
      f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                       vc_EVMaximumPowerLimit, w,
                                     cc_maximum_value_physicalValue_case4,
                                     v_vct);
      vc_isPowLimit := true;
}
if(ispresent(evChargeParameter.dC_EVChargeParameter.departureTime)) {
    vc_DepartureTime := evChargeParameter.dC_EVChargeParameter.departureTime;
    if(vc_DepartureTime == 0) {
       logWarn("A departure time equals to 0s might lead to interoperability " &
               "issues if the EVSE proceeds as defined in [V2G2-303].");
    }
}
log("------------------------------------------------------------");
log("EV maximum charging limits:");
log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumCurrentLimit)));
log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumVoltageLimit)));
if(vc_isPowLimit) {
    log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVMaximumPowerLimit)));
}
log("------------------------------------------------------------");

if(getverdict == pass) {
  setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
  v_sASchedules := f_getSchedules(vc_SAScheduleTupleIdRenegotiation,
                                  vc_SAScheduleTupleIdRenegotiation2,
                                  vc_SAScheduleTupleIdRenegotiation3,
                                  vc_maxEntriesSAScheduleTuple);
  vc_SASchedule_List := v_sASchedules.sAScheduleList;
  log("------------------------------------------------------------");
  log("EVSE maximum charging limits:");
  log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumCurrentLimit)));
  log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumVoltageLimit)));
  log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumPowerLimit)));
  log("EVSEMinimumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMinimumCurrentLimit)));
  log("EVSEMinimumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMinimumVoltageLimit)));
  log("------------------------------------------------------------");

  if(PICS_CMN_CMN_IdentificationMode == pnC and vc_useSalesTariffs) {

      vc_myChargeParameterBodyType := md_CMN_BodyType_001(
                        md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                        oK, finished, v_sASchedules,
                        md_EVCC_DC_EVSEChargeParameter_002(
                        vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                        vc_EVSEMaximumPowerLimit,
                        vc_EVSEMaximumVoltageLimit,
                        vc_EVSEMinimumCurrentLimit,
                        vc_EVSEMinimumVoltageLimit,
                        vc_EVSECurrentRegulationTolerance,
                        vc_EVSEPeakCurrentRipple,
                        vc_EVSEEnergyToBeDelivered)));

      vc_salesTariffIdDigestValueMap :=
              fx_getIdDigestMapFromScheduleListType(
              vc_SASchedule_List,
              par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                - cc_salesTariff_processingTime_remains);

            if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                oK, ongoing, omit,
                                md_EVCC_DC_EVSEChargeParameter_002(
                                vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                vc_EVSEMaximumPowerLimit,
                                vc_EVSEMaximumVoltageLimit,
                                vc_EVSEMinimumCurrentLimit,
                                vc_EVSEMinimumVoltageLimit,
                                omit,
                                vc_EVSEPeakCurrentRipple,
                                omit)))));
                v_isFinished := false;
            } else {

              var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
              f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                          vc_myChargeParameterBodyType,
                                          vc_salesTariffIdDigestValueMap));
              var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                     vc_privateKey);

              v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);
            }
          } else {
              v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
          }

          if(v_isFinished) {
                f_EVCC_changeValidStateCondition(e_StateB,vc_state);
              tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
              pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    v_messageHeader,
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                    oK, finished, v_sASchedules,
                                    md_EVCC_DC_EVSEChargeParameter_002(
                                    vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                    vc_EVSEMaximumPowerLimit,
                                    vc_EVSEMaximumVoltageLimit,
                                    vc_EVSEMinimumCurrentLimit,
                                    vc_EVSEMinimumVoltageLimit,
                                    vc_EVSECurrentRegulationTolerance,
                                    vc_EVSEPeakCurrentRipple,
                                    vc_EVSEEnergyToBeDelivered)))));
          }
          repeat;
      }
}
[vc_renegotiation and
(v_renegotiationCounter == 2 or
 v_isCableCheck == true)] a_EVCC_StateCorD(
                                pt_HAL_61851_Internal_Port, vc_state){
   v_stateNotDetected := false;
   repeat;
}
[vc_renegotiation and
(v_renegotiationCounter == 2 or
 v_isCableCheck == true)]
 pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                   md_CMN_BodyType_001(
                                md_CMN_DC_CableCheckReq_001(
                                md_CMN_DC_EVStatus_001(?,?,?)))))
                                -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        vc_DC_EVSEStatus.eVSEIsolationStatus := invalid;
        vc_DC_EVSEStatus.eVSEStatusCode := eVSE_IsolationMonitoringActive;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            v_cnt := v_cnt + 1;

            if(v_stateNotDetected){
                f_EVCC_StateChange_B2CorD(v_HAL_61851_Listener,
                                          par_EVCC_StateB_Shutdown_Timeout, v_vct);
                v_stateNotDetected := false;
            }

            if(v_cnt == 1) {
                t1.start;
            }

            if(v_renegotiationCounter == 2) {
               v_renegotiationCounter := v_renegotiationCounter + 1;
               v_isCableCheck := true;
            }
            tc_V2G_SECC_Sequence_Timer.stop;
            if(v_requestMessage.body.bodyElement.cableCheckReq.
               dC_EVStatus.eVReady == false) {
                logWarn("The field 'evReady' should be set to 'true'.");
            }


            setverdict(pass, "CableCheckReq message was correct.");

            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_CableCheckRes_001(
                                          oK, ongoing, vc_DC_EVSEStatus))));
            if(getverdict == pass) {
                repeat;
            }
        }
        [] t_cableCheck.timeout {
            v_cableCheckFinished := true;
            t1.start;
            repeat;
        }
        [] t1.timeout {
            if((not t_cableCheck.running) and (not v_cableCheckFinished)) {
                t_cableCheck.start(par_SECC_timeCableCheckVoltage);
            }
            if(not v_cableCheckFinished) {
                t1.start;
                repeat;

            } else {
                alt {
                    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(md_CMN_DC_CableCheckReq_001(
                                            md_CMN_DC_EVStatus_001(?,?,?)))))
                                            -> value v_requestMessage {

                        if(v_requestMessage.body.bodyElement.cableCheckReq.
                           dC_EVStatus.eVReady == false) {
                            logWarn("The field 'evReady' should be set to 'true'.");
                        }
                        tc_V2G_SECC_Sequence_Timer.stop;
                        setverdict(pass, "CableCheckReq message was correct.");
                    }
                    [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                                   "misleadingly terminated " &
                                                                   "by the SUT.");
                    [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                                    {cableCheckReq := ?});
                    [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                                 "but cannot be decoded.");
                    [] a_EVCC_Unexpected_Message_Content(v_vct, "Received CableCheckReq message, " &
                                                                "but with invalid data content.");
                    [] a_EVCC_Timeout(v_vct);
                }
                vc_DC_EVSEStatus.eVSEIsolationStatus := valid;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        vc_DC_EVSEStatus.eVSEStatusCode := eVSE_Ready;

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_CableCheckRes_001(
                                          oK, finished, vc_DC_EVSEStatus))));
        repeat;
    }
}
[vc_renegotiation and
(v_renegotiationCounter == 3 or
 v_isPreCharge == true)]
 pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_PreChargeReq_001(
                                  md_CMN_DC_EVStatus_001(?,?,?),?,?))))
                                  -> value v_requestMessage {

       tc_V2G_SECC_Sequence_Timer.stop;
       v_cnt_PC := v_cnt_PC + 1;
       vc_EVTargetVoltage := v_requestMessage.body.bodyElement.
                            preChargeReq.eVTargetVoltage;
       if(v_cnt_PC <= cc_preChargeVoltageRampUpSteps) {
         v_voltage_temp := v_voltage_temp + v_voltageStep;
         vc_EVSEPresentVoltage.value_ := v_voltage_temp;
       }
       if(v_cnt_PC > cc_preChargeVoltageRampUpSteps) {
         vc_EVSEPresentVoltage.value_ := float2int(f_CMN_PhysicalValue_GetValue(
                                         vc_EVTargetVoltage));
       }
       if(v_renegotiationCounter == 3) {
          v_renegotiationCounter := v_renegotiationCounter + 1;
          v_isPreCharge := true;
       }
       if(v_requestMessage.body.bodyElement.preChargeReq.dC_EVStatus.
          eVReady == false) {
           logWarn("The field 'evReady' should be set to 'true'.");
       }
       vc_EVTargetCurrent := v_requestMessage.body.bodyElement.
                             preChargeReq.eVTargetCurrent;
       f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                         cc_maximum_value_physicalValue_case1,
                                         v_vct);
       vc_EVSEPresentCurrent := vc_EVTargetCurrent;
       f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                         cc_maximum_value_physicalValue_case2,
                                         v_vct);


       if(getverdict == pass) {
           setverdict(pass, "PreChargeReq message was correct.");
       }

       tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
       pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                       md_CMN_BodyType_001(
                                       md_CMN_DC_PreChargeRes_001(
                                       oK, vc_DC_EVSEStatus,
                                       vc_EVSEPresentVoltage))));
       if(getverdict == pass) {
         repeat;
       }
}
[vc_renegotiation and v_renegotiationCounter == 4]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_PowerDeliveryReq_002(
                                  start_, ?, ?,
                                  md_CMN_DC_EVPowerDeliveryParameter(
                                  md_CMN_DC_EVStatus_001(?,?,?),*,false)))))
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;

            var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                                powerDeliveryReq.sAScheduleTupleID;

            vc_chargingProfile := v_requestMessage.body.bodyElement.
                                  powerDeliveryReq.chargingProfile;

            if((v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation) or
               (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation2) or
               (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation3)) {
                setverdict(pass, "Valid SAScheduleTupleId was " &
                                 "chosen from SUT.");
                vc_SAScheduleTupleId := v_SAScheduleTupleId;
            } else {
                    setverdict(v_vct, "Invalid SAScheduleTupleId was " &
                                      "chosen from SUT.");
            }

            if(getverdict == pass) {
                f_checkValidMaxPowerEntries(vc_chargingProfile, vc_SASchedule_List,
                                            vc_SAScheduleTupleId, v_vct);
            }
            if(getverdict == pass) {
                vc_renegotiation := false;
                v_renegotiationCounter := 0;
                v_isCableCheck := false;
                v_isPreCharge := false;
                if(v_requestMessage.body.bodyElement.powerDeliveryReq.
                     eVPowerDeliveryParameter.dC_EVPowerDeliveryParameter.
                     dC_EVStatus.eVReady == false) {
                   logWarn("The field 'evReady' should be set to 'true'.");
                }


                setverdict(pass, "PowerDeliveryReq message was correct.");

                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(
                                                  vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_PowerDeliveryRes_001(
                                                oK, md_CMN_DC_EVSEStatus_002(
                                                vc_DC_EVSEStatus)))));
                repeat;
            }
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                              md_CMN_BodyType_001(
                                              md_CMN_DC_CurrentDemandReq_001(
                                              md_CMN_DC_EVStatus_001(?,?,?),
                                              ?,*,*,*,*,?,*,*,?))))
                                              -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            tc_V2G_SECC_Msg_Timer.start(
            par_V2G_SECC_Msg_CurrentDemand_Performance_Time);
            vc_EVRESSSOC := v_requestMessage.body.bodyElement.currentDemandReq.
            dC_EVStatus.eVRESSSOC;
            vc_EVTargetCurrent := v_requestMessage.body.bodyElement.
                                  currentDemandReq.eVTargetCurrent;
            f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                             cc_maximum_value_physicalValue_case1,
                                             v_vct);
            vc_EVSEPresentCurrent := vc_EVTargetCurrent;
            vc_EVTargetVoltage := v_requestMessage.body.bodyElement.
                                  currentDemandReq.eVTargetVoltage;
            f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                             cc_maximum_value_physicalValue_case2,
                                             v_vct);
            if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                eVMaximumCurrentLimit)) {
     vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                                 currentDemandReq.eVMaximumCurrentLimit;
     f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                      vc_EVMaximumCurrentLimit, a,
                                      cc_maximum_value_physicalValue_case1,
                                      v_vct);
}
if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
                eVMaximumVoltageLimit)) {
     vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                                 currentDemandReq.eVMaximumVoltageLimit;
     f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                      vc_EVMaximumVoltageLimit, v,
                                      cc_maximum_value_physicalValue_case2,
                                      v_vct);
}
if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
                eVMaximumPowerLimit)) {
     vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                               currentDemandReq.eVMaximumPowerLimit;
     f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                      vc_EVMaximumPowerLimit, w,
                                      cc_maximum_value_physicalValue_case4,
                                      v_vct);
}
log("-------------------------------------------------------------");
log("EV maximum charging limits:");
log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumCurrentLimit)));
log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumVoltageLimit)));
if(vc_isPowLimit) {
    log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVMaximumPowerLimit)));
}
log("-------------------------------------------------------------");
if(v_requestMessage.body.bodyElement.currentDemandReq.
      dC_EVStatus.eVReady == false) {
    logWarn("The field 'evReady' should be set to 'true'.");
}
v_loopCounter := v_loopCounter + 1;
log("-------------------------------------------------------------");
log("Number of received CurrentDemandReq messages: " &
       toString<integer>(v_loopCounter));
log("EVRESSSOC: " & toString<PercentValueType>(vc_EVRESSSOC) & "%.");
log("-------------------------------------------------------------");

if(v_loopCounter == PICS_CMN_CMN_LoopCounter) {
         vc_DC_EVSEStatus.eVSENotification := stopCharging;
         vc_DC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
}
if(v_loopCounter == PICS_CMN_CMN_RenegotiationLoopIndication and
   vc_renegotiation) {
         vc_DC_EVSEStatus.eVSENotification := reNegotiation;
         vc_DC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
}

if(vc_DC_EVSEStatus.eVSENotification == stopCharging or
   vc_DC_EVSEStatus.eVSENotification == reNegotiation) {
   if(not t_stopProcessing.running) {
       t_stopProcessing.start(par_EVCC_StopCharging_Timeout);
   }
}
if(not tc_V2G_SECC_Msg_Timer.running) {
    setverdict(inconc,"Msg performance time could not be fulfilled. " &
                      "The internal processing time should be optimized.");
} else {tc_V2G_SECC_Msg_Timer.stop;}
if(getverdict == pass) {
  setverdict(pass,"CurrentDemandReq message was correct. " &
                  "Waiting for PowerDeliveryReq message");
  log("-------------------------------------------------------------");
  log("EVSE maximum charging limits:");
  log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
     vc_EVSEMaximumCurrentLimit)));
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
            log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
               vc_EVSEMaximumVoltageLimit)));
            log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
               vc_EVSEMaximumPowerLimit)));
            log("------------------------------------------------------------");
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            // repeat sending current demand response message
            if(PICS_CMN_CMN_IdentificationMode == pnC) {
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                   md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                                   vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                                   false, false, false, vc_EVSEMaximumVoltageLimit,
                                   vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                                   vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, false))));
            } else {
               pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                                  vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                                  false, false, false, vc_EVSEMaximumVoltageLimit,
                                  vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                                  vc_EvseId, vc_SAScheduleTupleId, omit, false))));
            }
            repeat;
          }
      }

      [] t_stopProcessing.timeout {
          setverdict(fail, "SUT did not react to EVSENotification 'stopCharging' or 'renegotiate' " &
                           "within 'par_EVCC_StopCharging_Timeout'.");
      }
      [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                     "terminated by the SUT.");
      [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                  {currentDemandReq := ?},
                                                  {powerDeliveryReq := ?});
      [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                   "but cannot be decoded.");
      [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                  "CurrentDemandReq message, " &
                                                  "but with invalid data content.");
      [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
  }

  function f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_002(in HAL_61851_Listener
                                                              v_HAL_61851_Listener,
                                                              in verdicttype v_vct)
                                                              runs on EVCC_Tester
                                                              return verdicttype {

      var charstring v_message;
      var V2G_Message v_requestMessage;
      var integer v_loopCounter := 1;
      var boolean v_isMetering := false;
      var integer v_renegotiationCounter := 0;
      var boolean v_isCableCheck := false;
      var boolean v_isPreCharge := false;
      var boolean isSignatureOk := true;
      var MessageHeaderType v_messageHeader;
      var charstring v_id;
      var SASchedules_group v_sASchedules;
      var boolean v_isFinished := true;
      timer t1 := 0.1;
      timer t_cableCheck;
      timer t_keepVoltage := 3.0;
      var boolean v_isolationCheck := false;
      var boolean v_cableCheckFinished := false;
      var boolean v_stateNotDetected := true;
      var integer v_cnt := 0;
      var integer v_voltage_temp := 0;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var integer v_cnt_PC := 0;
timer t_stop;
var timer t_stopProcessing;
var float v_processingDummy := 5.0;

var integer v_voltageStep := float2int(f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage))/
                                    cc_preChargeVoltageRampUpSteps;

log("------------------------------------------------------------");
log("Number of received CurrentDemandReq messages: " & toString<integer>(v_loopCounter));
log("------------------------------------------------------------");
if(not tc_V2G_SECC_Msg_Timer.running) {
    setverdict(inconc,"Msg performance time could not be fulfilled. " &
                      "The internal processing time should be optimized.");
} else {tc_V2G_SECC_Msg_Timer.stop;}
log("------------------------------------------------------------");
log("EVSE maximum charging limits:");
log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumCurrentLimit)));
log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumVoltageLimit)));
log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumPowerLimit)));
log("------------------------------------------------------------");
tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                 md_CMN_MessageHeaderType_001(vc_SessionID),
                 md_CMN_BodyType_001(
                 md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                 vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                 false, false, false, vc_EVSEMaximumVoltageLimit,
                 vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                 vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, true))));

alt {
  [v_isMetering and not vc_renegotiation]
  pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                mdw_CMN_DC_PowerDeliveryReq_003(
                                stop_, vc_SAScheduleTupleId, *))))
                                -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        tc_CMN_ProcessingTime_Timer.start(v_processingDummy);
        setverdict(pass, "PowerDeliveryReq message was correct.");
        vc_processingTime := tc_CMN_ProcessingTime_Timer.read;
        tc_CMN_ProcessingTime_Timer.stop;
  }
  [v_isMetering and
   v_renegotiationCounter == 0]
   pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_PowerDeliveryReq_002(
                                renegotiate, vc_SAScheduleTupleId, *,
                                md_CMN_DC_EVPowerDeliveryParameter(
                                md_CMN_DC_EVStatus_001(?,?,?),*,?))))) {

        tc_V2G_SECC_Sequence_Timer.stop;
        t_stopProcessing.stop;
        vc_renegotiation := true;
        v_renegotiationCounter := v_renegotiationCounter + 1;


        setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                        "be executed.");

            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_PowerDeliveryRes_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    oK, md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

            f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener,
                                par_V2G_SECC_Sequence_Timeout, v_vct);
        vc_DC_EVSEStatus.eVSENotification := none_;
        repeat;
    }
    [vc_renegotiation and v_renegotiationCounter == 1]
    pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_ChargeParamterDiscoveryReq_001(*,?,?))))
                                -> value v_requestMessage {

        v_isFinished := true;
        vc_DC_EVSEStatus := {par_EVCC_NotificationMaxDelay, none_, invalid, eVSE_Ready};
        tc_V2G_SECC_Sequence_Timer.stop;
        if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                    maxEntriesSAScheduleTuple)) {
            vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                            chargeParameterDiscoveryReq.
                                            maxEntriesSAScheduleTuple;
        }
        v_renegotiationCounter := v_renegotiationCounter + 1;
        var EVChargeParameter_group evChargeParameter := v_requestMessage.body.bodyElement.
                                                         chargeParameterDiscoveryReq.
                                                         eVChargeParameter;
        if(evChargeParameter.dC_EVChargeParameter.dC_EVStatus.eVReady == true) {
            logWarn("The field 'evReady' should be set to 'false'.");
        }
        vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                        chargeParameterDiscoveryReq.eVChargeParameter.dC_EVChargeParameter.
                        eVMaximumCurrentLimit;
        f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                         vc_EVMaximumCurrentLimit, a,
                                       cc_maximum_value_physicalValue_case1,
                                       v_vct);
        vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                          chargeParameterDiscoveryReq.eVChargeParameter.dC_EVChargeParameter.
                          eVMaximumVoltageLimit;
        f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                         vc_EVMaximumVoltageLimit, v,
                                       cc_maximum_value_physicalValue_case2,
                                       v_vct);
        if(ispresent(v_requestMessage.body.bodyElement.
                    chargeParameterDiscoveryReq.eVChargeParameter.dC_EVChargeParameter.
                    eVMaximumPowerLimit)) {
            vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                      chargeParameterDiscoveryReq.eVChargeParameter.
                                      dC_EVChargeParameter.eVMaximumPowerLimit;
            f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                             vc_EVMaximumPowerLimit, w,
                                           cc_maximum_value_physicalValue_case4,
                                           v_vct);
            vc_isPowLimit := true;
        }
        log("-------------------------------------------------------------");
        log("EV maximum charging limits:");
        log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumCurrentLimit)));
        log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumVoltageLimit)));
        if(vc_isPowLimit) {
            log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumPowerLimit)));
        }
        log("-------------------------------------------------------------");
        if(ispresent(evChargeParameter.dC_EVChargeParameter.departureTime)) {
            vc_DepartureTime := evChargeParameter.dC_EVChargeParameter.departureTime;
            if(vc_DepartureTime == 0) {
               logWarn("A departure time equals to 0s might lead to interoperability " &
                       "issues if the EVSE proceeds as defined in [V2G2-303].");
            }
        }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
if(getverdict == pass) {
  setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
  v_sASchedules := f_getSchedules(vc_SAScheduleTupleIdRenegotiation,
                                  vc_SAScheduleTupleIdRenegotiation2,
                                  vc_SAScheduleTupleIdRenegotiation3,
                                  vc_maxEntriesSAScheduleTuple);
  vc_SASchedule_List := v_sASchedules.sAScheduleList;
  log("------------------------------------------------------------");
  log("EVSE maximum charging limits:");
  log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumCurrentLimit)));
  log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumVoltageLimit)));
  log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumPowerLimit)));
  log("EVSEMinimumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMinimumCurrentLimit)));
  log("EVSEMinimumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMinimumVoltageLimit)));
  log("------------------------------------------------------------");

  if(vc_useSalesTariffs) {

      vc_myChargeParameterBodyType := md_CMN_BodyType_001(
                       md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                       oK, finished, v_sASchedules,
                       md_EVCC_DC_EVSEChargeParameter_002(
                       vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                       vc_EVSEMaximumPowerLimit,
                       vc_EVSEMaximumVoltageLimit,
                       vc_EVSEMinimumCurrentLimit,
                       vc_EVSEMinimumVoltageLimit,
                       vc_EVSECurrentRegulationTolerance,
                       vc_EVSEPeakCurrentRipple,
                       vc_EVSEEnergyToBeDelivered)));

      vc_salesTariffIdDigestValueMap :=
                fx_getIdDigestMapFromScheduleListType(
                vc_SASchedule_List,
                par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
                - cc_salesTariff_processingTime_remains);

      if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
          tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
          pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                 oK, ongoing, omit,
                                 md_EVCC_DC_EVSEChargeParameter_002(
                                 vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                 vc_EVSEMaximumPowerLimit,
                                 vc_EVSEMaximumVoltageLimit,
                                 vc_EVSEMinimumCurrentLimit,
                                 vc_EVSEMinimumVoltageLimit,
                                 omit,
                                 vc_EVSEPeakCurrentRipple,
                                 omit)))));
          v_isFinished := false;
      } else {

        var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
        f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                              vc_myChargeParameterBodyType,
                                              vc_salesTariffIdDigestValueMap));
        var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                         vc_privateKey);

        v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);
      }
    } else {
        v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
    }

    if(v_isFinished) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                f_EVCC_changeValidStateCondition(e_StateB,vc_state);
              tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
              pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                          v_messageHeader,
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                          oK, finished, v_sASchedules,
                                          md_EVCC_DC_EVSEChargeParameter_002(
                                          vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                          vc_EVSEMaximumPowerLimit,
                                          vc_EVSEMaximumVoltageLimit,
                                          vc_EVSEMinimumCurrentLimit,
                                          vc_EVSEMinimumVoltageLimit,
                                          vc_EVSECurrentRegulationTolerance,
                                          vc_EVSEPeakCurrentRipple,
                                          vc_EVSEEnergyToBeDelivered)))));
            }
            repeat;
        }
    }
    [vc_renegotiation and
    (v_renegotiationCounter == 2 or
     v_isCableCheck == true)] a_EVCC_StateCorD(
                                  pt_HAL_61851_Internal_Port, vc_state){
       v_stateNotDetected := false;
       repeat;
    }
    [vc_renegotiation and
    (v_renegotiationCounter == 2 or
     v_isCableCheck == true)]
     pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_CableCheckReq_001(
                                  md_CMN_DC_EVStatus_001(?,?,?)))))
                                  -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            vc_DC_EVSEStatus.eVSEIsolationStatus := invalid;
            vc_DC_EVSEStatus.eVSEStatusCode := eVSE_IsolationMonitoringActive;
            v_cnt := v_cnt + 1;

            if(v_stateNotDetected){
                f_EVCC_StateChange_B2CorD(v_HAL_61851_Listener,
                                  par_EVCC_StateB_Shutdown_Timeout, v_vct);
                v_stateNotDetected := false;
            }


            if(v_renegotiationCounter == 2) {
               v_renegotiationCounter := v_renegotiationCounter + 1;
               v_isCableCheck := true;
            }
            tc_V2G_SECC_Sequence_Timer.stop;
            if(v_requestMessage.body.bodyElement.cableCheckReq.
               dC_EVStatus.eVReady == false) {
                logWarn("The field 'evReady' should be set to 'true'.");
            }


            setverdict(pass, "CableCheckReq message was correct.");

            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_CableCheckRes_001(
                                  oK, ongoing, vc_DC_EVSEStatus))));
            if(getverdict == pass) {
                repeat;
            }
    }
    [] t_cableCheck.timeout {
        v_cableCheckFinished := true;
        t1.start;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        repeat;
    }
    [] t1.timeout {
            if((not t_cableCheck.running) and (not v_cableCheckFinished)) {
                t_cableCheck.start(par_SECC_timeCableCheckVoltage);
            }
        if(not v_cableCheckFinished) {
            t1.start;
            repeat;

        } else {
            alt {
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(md_CMN_DC_CableCheckReq_001(
                                                md_CMN_DC_EVStatus_001(?,?,?)))))
                                                -> value v_requestMessage {

                    if(v_requestMessage.body.bodyElement.cableCheckReq.
                       dC_EVStatus.eVReady == false) {
                        logWarn("The field 'evReady' should be set to 'true'.");
                    }
                    tc_V2G_SECC_Sequence_Timer.stop;
                    setverdict(pass, "CableCheckReq message was correct.");
                }
                [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP connection was " &
                                                              "misleadingly terminated " &
                                                              "by the SUT.");
                [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                                {cableCheckReq := ?});
                [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                             "but cannot be decoded.");
                [] a_EVCC_Unexpected_Message_Content(v_vct, "Received CableCheckReq message, " &
                                                            "but with invalid data content.");
                [] a_EVCC_Timeout(v_vct);
            }
            vc_DC_EVSEStatus.eVSEIsolationStatus := valid;
            vc_DC_EVSEStatus.eVSEStatusCode := eVSE_Ready;

            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_DC_CableCheckRes_001(
                                                oK, finished, vc_DC_EVSEStatus))));
            repeat;
        }
    }
    [vc_renegotiation and
    (v_renegotiationCounter == 3 or
     v_isPreCharge == true)]
     pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_PreChargeReq_001(
                                    md_CMN_DC_EVStatus_001(?,?,?),?,?))))
                                    -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        v_cnt_PC := v_cnt_PC + 1;
        vc_EVTargetVoltage := v_requestMessage.body.bodyElement.
                              preChargeReq.eVTargetVoltage;
        if(v_cnt_PC <= cc_preChargeVoltageRampUpSteps) {
          v_voltage_temp := v_voltage_temp + v_voltageStep;
          vc_EVSEPresentVoltage.value_ := v_voltage_temp;
        }
        if(v_cnt_PC > cc_preChargeVoltageRampUpSteps) {
          vc_EVSEPresentVoltage.value_ := float2int(f_CMN_PhysicalValue_GetValue(
                                          vc_EVTargetVoltage));
        }
        if(v_renegotiationCounter == 3) {
           v_renegotiationCounter := v_renegotiationCounter + 1;
           v_isPreCharge := true;
        }
        if(v_requestMessage.body.bodyElement.preChargeReq.dC_EVStatus.
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            eVReady == false) {
             logWarn("The field 'evReady' should be set to 'true'.");
         }
         vc_EVTargetCurrent := v_requestMessage.body.bodyElement.
                               preChargeReq.eVTargetCurrent;
         f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                          cc_maximum_value_physicalValue_case1,
                                          v_vct);
         vc_EVSEPresentCurrent := vc_EVTargetCurrent;
         f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                          cc_maximum_value_physicalValue_case2,
                                          v_vct);

         if(getverdict == pass) {
            setverdict(pass, "PreChargeReq message was correct.");
         }

         tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
         pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_DC_PreChargeRes_001(
                                      oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage))));
         if(getverdict == pass) {
           repeat;
         }
      }
      [vc_renegotiation and v_renegotiationCounter == 4]
      pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     md_CMN_DC_PowerDeliveryReq_002(
                                     start_, ?, ?,
                                     md_CMN_DC_EVPowerDeliveryParameter(
                                     md_CMN_DC_EVStatus_001(?,?,?),*,false)))))
                                     -> value v_requestMessage {

      tc_V2G_SECC_Sequence_Timer.stop;

      var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                          powerDeliveryReq.sAScheduleTupleID;

      vc_chargingProfile := v_requestMessage.body.bodyElement.
                            powerDeliveryReq.chargingProfile;

      if((v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation) or
         (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation2) or
         (v_SAScheduleTupleId == vc_SAScheduleTupleIdRenegotiation3)) {
          setverdict(pass, "Valid SAScheduleTupleId was " &
                           "chosen from SUT.");
      } else {
              setverdict(v_vct, "Invalid SAScheduleTupleId was " &
                                "chosen from SUT.");
      }
      vc_SAScheduleTupleId := v_SAScheduleTupleId;

          if(getverdict == pass) {
          f_checkValidMaxPowerEntries(vc_chargingProfile, vc_SASchedule_List,
                                      vc_SAScheduleTupleId, v_vct);
      }
      if(getverdict == pass) {
          vc_renegotiation := false;
          v_renegotiationCounter := 0;
          v_isCableCheck := false;
          v_isPreCharge := false;
          if(v_requestMessage.body.bodyElement.powerDeliveryReq.
             eVPowerDeliveryParameter.dC_EVPowerDeliveryParameter.
             dC_EVStatus.eVReady == false) {
             logWarn("The field 'evReady' should be set to 'true'.");
          }

          setverdict(pass, "PowerDeliveryReq message was correct.");

          tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
          pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_MessageHeaderType_001(
                                        vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_PowerDeliveryRes_001(
                                        oK, md_CMN_DC_EVSEStatus_002(
                                        vc_DC_EVSEStatus)))));
            repeat;
        }
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_CurrentDemandReq_001(
                                    md_CMN_DC_EVStatus_001(?,?,?),?,*,*,*,*,?,*,*,?))))
                                    -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        tc_V2G_SECC_Msg_Timer.start(
        par_V2G_SECC_Msg_CurrentDemand_Performance_Time);
        vc_EVRESSSOC := v_requestMessage.body.bodyElement.currentDemandReq.
        dC_EVStatus.eVRESSSOC;
        vc_EVTargetCurrent := v_requestMessage.body.bodyElement.currentDemandReq.
                              eVTargetCurrent;
        f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                         cc_maximum_value_physicalValue_case1,
                                         v_vct);
        vc_EVSEPresentCurrent := vc_EVTargetCurrent;
        vc_EVTargetVoltage := v_requestMessage.body.bodyElement.currentDemandReq.
                              eVTargetVoltage;
        f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                         cc_maximum_value_physicalValue_case2,
                                         v_vct);
        if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
                     eVMaximumCurrentLimit)) {
             vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                                         currentDemandReq.eVMaximumCurrentLimit;
             f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                              vc_EVMaximumCurrentLimit, a,
                                              cc_maximum_value_physicalValue_case1,
                                              v_vct);
        }
        if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
                     eVMaximumVoltageLimit)) {
             vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                                         currentDemandReq.eVMaximumVoltageLimit;
             f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                              vc_EVMaximumVoltageLimit, v,
                                              cc_maximum_value_physicalValue_case2,
                                              v_vct);
        }
        if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
                     eVMaximumPowerLimit)) {
             vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                       currentDemandReq.eVMaximumPowerLimit;
             f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                              vc_EVMaximumPowerLimit, w,
                                              cc_maximum_value_physicalValue_case4,
                                              v_vct);
        }
        log("-------------------------------------------------------------");
        log("EV maximum charging limits:");
        log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumCurrentLimit)));
        log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumVoltageLimit)));
        if(vc_isPowLimit) {
            log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumPowerLimit)));
        }
        log("-------------------------------------------------------------");
        if(v_requestMessage.body.bodyElement.currentDemandReq.dC_EVStatus.
            eVReady == false) {
            logWarn("The field 'evReady' should be set to 'true'.");
        }
        if(not v_isMetering) {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            setverdict(v_vct,"The SUT did not send a Metering Receipt message.");
        }
        v_loopCounter := v_loopCounter + 1;
        log("------------------------------------------------------------");
        log("Number of received CurrentDemandReq messages: " & toString<integer>(v_loopCounter));
        log("EVRESSSOC: " & toString<PercentValueType>(vc_EVRESSSOC) & "%.");
        log("------------------------------------------------------------");

        if(v_loopCounter == PICS_CMN_CMN_LoopCounter) {
                vc_DC_EVSEStatus.eVSENotification := stopCharging;
                vc_DC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
        }
        if(v_loopCounter == PICS_CMN_CMN_RenegotiationLoopIndication and
           vc_renegotiation) {
                vc_DC_EVSEStatus.eVSENotification := reNegotiation;
                vc_DC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
        }

        if(vc_DC_EVSEStatus.eVSENotification == stopCharging or
           vc_DC_EVSEStatus.eVSENotification == reNegotiation) {
           if(not t_stopProcessing.running) {
               t_stopProcessing.start(par_EVCC_StopCharging_Timeout);
           }
        }
        if(not tc_V2G_SECC_Msg_Timer.running) {
            setverdict(inconc,"Msg performance time could not be fulfilled. " &
                              "The internal processing time should be optimized.");
        } else {tc_V2G_SECC_Msg_Timer.stop;}
        if(getverdict == pass) {
            setverdict(pass,"CurrentDemandReq message was correct. Waiting for " &
                            "PowerDeliveryReq message");
           log("------------------------------------------------------------");
           log("EVSE maximum charging limits:");
           log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
               vc_EVSEMaximumCurrentLimit)));
           log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
               vc_EVSEMaximumVoltageLimit)));
           log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
               vc_EVSEMaximumPowerLimit)));
           log("------------------------------------------------------------");
           tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
           // repeat sending current demand response message
           pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                             md_CMN_MessageHeaderType_001(vc_SessionID),
                             md_CMN_BodyType_001(
                             md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                             vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                             false, false, false, vc_EVSEMaximumVoltageLimit,
                             vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                             vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, true))));

           repeat;
        }
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_003(vc_SessionID,
                                    md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_MeteringReceiptReq_001(
                                    *, vc_SessionID, *, vc_MeterInfo))))
                                    -> value v_requestMessage{

        tc_V2G_SECC_Sequence_Timer.stop;
        v_isMetering := true;
        var Signature authSignature := v_requestMessage.header.signature_;
        var BodyType meteringReceiptReqBodyType := v_requestMessage.body;

        f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                    cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

        if(getverdict == pass) {

                isSignatureOk := f_verifySignature(meteringReceiptReqBodyType,
                                                   authSignature,
                                                   vc_contractSignatureCertChain.
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                    x509Certificate, v_vct);

            if(isSignatureOk) {
               setverdict(pass,"MeteringReceiptReq message was correct.");
               tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
               pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_DC_MeteringReceiptRes_001(
                                          oK,vc_DC_EVSEStatus))));
               repeat;
            }
            else {
                setverdict(v_vct, "Invalid signature of the MeteringReceiptReq " &
                                  "message was received.");
            }
         }
    }

    [] t_stopProcessing.timeout {
        setverdict(fail, "SUT did not react to EVSENotification " &
                         "'stopCharging' or 'renegotiate' " &
                         "within 'par_EVCC_StopCharging_Timeout'.");
    }
    [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                   "terminated by the SUT.");
    [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var3(v_vct, "Unexpected V2G message was received.",
                                                {meteringReceiptReq := ?},
                                                {currentDemandReq := ?},
                                                {powerDeliveryReq := ?});
    [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                 "but cannot be decoded.");
    [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                "CurrentDemandReq message, " &
                                                "but with invalid data content.");
    [] a_EVCC_Timeout(v_vct);
  }
  return getverdict;
}

function f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_003(urn_iso_15118_2_2013_MsgDataTypes.
                                                            ResponseCodeType responseCode,
                                                            in HAL_61851_Listener
                                                            v_HAL_61851_Listener)
                                                            runs on EVCC_Tester
                                                            return verdicttype  {

    var charstring v_message;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

      f_EVCC_setPwmMode(e_PosVolt12);
      f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_PowerDeliveryRes_001(
                                responseCode, md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

    tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
           isStateB := true;
           tc_EVCC_StateB_Shutdown_Timer.stop;
           if(isClosedTCP == false) {repeat;}
        }
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.") {
           isClosedTCP := true;
           tc_CMN_TCP_Connection_Termination_Timer.stop;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            if((isStateB == false)) {repeat;}
        }
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_SessionStopReq_001(?)))) {
            tc_V2G_SECC_Sequence_Timer.stop;
            setverdict(fail,"Received SessionStopReq message was not expected.");
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout() {
           v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
           if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
        }
        [] a_EVCC_StateB_Shutdown_Timeout() {
           v_charstring := v_charstring & "SUT has not changed to CP State B " &
                           "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
           if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
        }
    }
    if(not(isStateB and isClosedTCP) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_004(in HAL_61851_Listener
                                                            v_HAL_61851_Listener)
                                                            runs on EVCC_Tester
                                                            return verdicttype  {

    timer t_wait;
    var float v_timeShift := 1.0;
    var boolean isStateB := false;
    var boolean isClosedTCP := false;
    var charstring v_charstring := "";

    // Wait until SUT message timeout
    log("Proccessing time: " & toString<float>(vc_processingTime) & "s.");
    t_wait.start(par_V2G_EVCC_Msg_Timeout_PowerDeliveryReq -
                 v_timeShift - vc_processingTime);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

          f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        sleep(v_timeShift - par_CMN_Transmission_Delay);

        tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
               isStateB := true;
               tc_EVCC_StateB_Shutdown_Timer.stop;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                if(isClosedTCP == false) {repeat;}
            }
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
                isClosedTCP := true;
                tc_CMN_TCP_Connection_Termination_Timer.stop;
                if((isStateB == false)) {repeat;}
            }
            [] a_EVCC_CMN_SessionStop_Termination() {
                 tc_CMN_TCP_Connection_Termination_Timer.start(
                 par_CMN_TCP_Connection_Termination_Timeout);
                 repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
                v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                                "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
                if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
            }
            [] a_EVCC_StateB_Shutdown_Timeout() {
                v_charstring := v_charstring & "SUT has not changed to CP State B " &
                                "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
                if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
            }
        }
        if(not(isStateB and isClosedTCP) and v_charstring != "") {
           setverdict(fail,v_charstring);
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_005(in verdicttype v_vct)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

  var V2G_Message v_requestMessage;
  var integer v_loopCounter := 0;
  var boolean v_renegotiation := false;
  timer t_stop;
  var timer t_stopProcessing;
  log("-------------------------------------------------------------");
  log("EVSE maximum charging limits:");
  log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumCurrentLimit)));
  log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumVoltageLimit)));
  log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumPowerLimit)));
  log("-------------------------------------------------------------");

  tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
  if(PICS_CMN_CMN_IdentificationMode == pnC) {
      pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                        md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                        vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                        false, false, false, vc_EVSEMaximumVoltageLimit,
                        vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                        vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, false))));
  } else {
     pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                       md_CMN_MessageHeaderType_001(vc_SessionID),
                       md_CMN_BodyType_001(
                       md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                       vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                       false, false, false, vc_EVSEMaximumVoltageLimit,
                       vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                       vc_EvseId, vc_SAScheduleTupleId, omit, false))));
  }

  alt {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
[v_renegotiation]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_PowerDeliveryReq_002(
                                renegotiate, vc_SAScheduleTupleId, *,
                                md_CMN_DC_EVPowerDeliveryParameter(
                                md_CMN_DC_EVStatus_001(?,?,?),*,?)))))
                                -> value v_requestMessage {

       tc_V2G_SECC_Sequence_Timer.stop;
       setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                       "be executed.");
}
[] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_CurrentDemandReq_001(
                                md_CMN_DC_EVStatus_001(?,?,?),?,*,*,*,*,?,*,*,?))))
                                -> value v_requestMessage {

     tc_V2G_SECC_Sequence_Timer.stop;
     vc_EVTargetCurrent := v_requestMessage.body.bodyElement.currentDemandReq.
                           eVTargetCurrent;
     f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                       cc_maximum_value_physicalValue_case1,
                                      v_vct);
     vc_EVSEPresentCurrent := vc_EVTargetCurrent;
     vc_EVTargetVoltage := v_requestMessage.body.bodyElement.currentDemandReq.
                           eVTargetVoltage;
     f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                       cc_maximum_value_physicalValue_case2,
                                      v_vct);

     if(ispresent(v_requestMessage.body.bodyElement.
                  currentDemandReq.eVMaximumCurrentLimit)) {
        vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                                    currentDemandReq.eVMaximumCurrentLimit;
        f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                         vc_EVMaximumCurrentLimit, a,
                                         cc_maximum_value_physicalValue_case1,
                                         v_vct);
     }
     if(ispresent(v_requestMessage.body.bodyElement.
                  currentDemandReq.eVMaximumVoltageLimit)) {
          vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                                      currentDemandReq.eVMaximumVoltageLimit;
          f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                           vc_EVMaximumVoltageLimit, v,
                                           cc_maximum_value_physicalValue_case2,
                                           v_vct);
     }
     if(ispresent(v_requestMessage.body.bodyElement.
                  currentDemandReq.eVMaximumPowerLimit)) {
          vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                    currentDemandReq.eVMaximumPowerLimit;
          f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                           vc_EVMaximumPowerLimit, w,
                                           cc_maximum_value_physicalValue_case4,
                                           v_vct);
     }
     log("-------------------------------------------------------------");
     log("EV maximum charging limits:");
     log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
         vc_EVMaximumCurrentLimit)));
     log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
         vc_EVMaximumVoltageLimit)));
     if(vc_isPowLimit) {
         log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
             vc_EVMaximumPowerLimit)));
     }
     log("-------------------------------------------------------------");
     if(getverdict == pass) {
        setverdict(pass,"CurrentDemandReq message was correct.");
     }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        if(v_loopCounter == 0) {
                v_renegotiation := true;
                vc_DC_EVSEStatus.eVSENotification := reNegotiation;
                vc_DC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
        }


           if(t_stop.running) {
               t_stop.stop;
           }
        log("-------------------------------------------------------------");
        log("EVSE maximum charging limits:");
        log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumCurrentLimit)));
        log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumVoltageLimit)));
        log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumPowerLimit)));
        log("-------------------------------------------------------------");
        v_loopCounter := v_loopCounter + 1;
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        // repeat sending current demand response message
        if(PICS_CMN_CMN_IdentificationMode == pnC) {
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                                vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                                false, false, false, vc_EVSEMaximumVoltageLimit,
                                vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                                vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, false))));
        } else {
           pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                               md_CMN_MessageHeaderType_001(vc_SessionID),
                               md_CMN_BodyType_001(
                               md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                               vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                               false, false, false, vc_EVSEMaximumVoltageLimit,
                               vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                               vc_EvseId, vc_SAScheduleTupleId, omit, false))));
        }
        if(getverdict == pass) {
           repeat;
        }
    }

    [] t_stopProcessing.timeout {
        setverdict(fail, "SUT did not react to EVSENotification 'renegotiate' " &
                         "within 'par_EVCC_StopCharging_Timeout'.");
    }
    [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                   "terminated by the SUT.");
    [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                {currentDemandReq := ?},
                                                {powerDeliveryReq := ?});
    [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                 "but cannot be decoded.");
    [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                "CurrentDemandReq message, " &
                                                "but with invalid data content.");
    [] a_EVCC_Timeout(v_vct);
  }
  return getverdict;
}

function f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_006(in verdicttype v_vct)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

  var V2G_Message v_requestMessage;
  var integer v_loopCounter := 0;
  var boolean v_isMetering := false;
  var boolean v_renegotiation := false;
  var boolean isSignatureOk := true;
  timer t_stop;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
var timer t_stopProcessing;

tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                              md_CMN_BodyType_001(
                              md_EVCC_DC_MeteringReceiptRes_001(
                              oK,vc_DC_EVSEStatus))));


alt {
  [v_renegotiation and v_isMetering]
  pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_PowerDeliveryReq_002(
                                  renegotiate, vc_SAScheduleTupleId, *,
                                  md_CMN_DC_EVPowerDeliveryParameter(
                                  md_CMN_DC_EVStatus_001(?,?,?),*,?)))))
                                  -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                        "be executed.");
  }
  [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_CurrentDemandReq_001(
                                  md_CMN_DC_EVStatus_001(?,?,?),?,*,*,*,*,?,*,*,?))))
                                  -> value v_requestMessage {

      tc_V2G_SECC_Sequence_Timer.stop;
      vc_EVTargetCurrent := v_requestMessage.body.bodyElement.currentDemandReq.
                            eVTargetCurrent;
      f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                       cc_maximum_value_physicalValue_case1,
                                      v_vct);
      vc_EVSEPresentCurrent := vc_EVTargetCurrent;
      vc_EVTargetVoltage := v_requestMessage.body.bodyElement.currentDemandReq.
                            eVTargetVoltage;
      f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                       cc_maximum_value_physicalValue_case2,
                                      v_vct);

      if(ispresent(v_requestMessage.body.bodyElement.
                   currentDemandReq.eVMaximumCurrentLimit)) {
         vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                                     currentDemandReq.eVMaximumCurrentLimit;
         f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                          vc_EVMaximumCurrentLimit, a,
                                          cc_maximum_value_physicalValue_case1,
                                          v_vct);
      }
      if(ispresent(v_requestMessage.body.bodyElement.
                   currentDemandReq.eVMaximumVoltageLimit)) {
           vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                                       currentDemandReq.eVMaximumVoltageLimit;
           f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                            vc_EVMaximumVoltageLimit, v,
                                          cc_maximum_value_physicalValue_case2,
                                          v_vct);
      }
      if(ispresent(v_requestMessage.body.bodyElement.
                   currentDemandReq.eVMaximumPowerLimit)) {
           vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                     currentDemandReq.eVMaximumPowerLimit;
           f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                            vc_EVMaximumPowerLimit, w,
                                          cc_maximum_value_physicalValue_case4,
                                          v_vct);
      }
      log("-------------------------------------------------------------");
      log("EV maximum charging limits:");
      log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            vc_EVMaximumCurrentLimit)));
        log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumVoltageLimit)));
        if(vc_isPowLimit) {
            log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumPowerLimit)));
        }
        log("-------------------------------------------------------------");
        if(getverdict == pass) {
           setverdict(pass,"CurrentDemandReq message was correct.");
        }

        if(v_loopCounter == 0) {
                vc_DC_EVSEStatus.eVSENotification := reNegotiation;
                vc_DC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
        }


           if(t_stop.running) {
               t_stop.stop;
           }
        log("-------------------------------------------------------------");
        log("EVSE maximum charging limits:");
        log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumCurrentLimit)));
        log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumVoltageLimit)));
        log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumPowerLimit)));
        log("-------------------------------------------------------------");
        v_loopCounter := v_loopCounter + 1;
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        // repeat sending current demand response message
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                          md_CMN_MessageHeaderType_001(vc_SessionID),
                          md_CMN_BodyType_001(
                          md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                          vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                          false, false, false, vc_EVSEMaximumVoltageLimit,
                          vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                          vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, true))));
        if(getverdict == pass) {
           repeat;
        }
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_003(vc_SessionID,
                                     md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                     md_CMN_BodyType_001(
                                     md_EVCC_CMN_MeteringReceiptReq_001(
                                     *, vc_SessionID, *, vc_MeterInfo))))
                                     -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        v_isMetering := true;
        if(vc_DC_EVSEStatus.eVSENotification == reNegotiation) {
           v_renegotiation := true;
        }
        var Signature authSignature := v_requestMessage.header.signature_;
        var BodyType meteringReceiptReqBodyType := v_requestMessage.body;

        f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                    cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

        if(getverdict == pass) {

                isSignatureOk := f_verifySignature(meteringReceiptReqBodyType,
                                                   authSignature,
                                                   vc_contractSignatureCertChain.
                                                   x509Certificate, v_vct);

            if(isSignatureOk) {
               setverdict(pass,"MeteringReceiptReq message was correct.");
               tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
               pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                                    md_CMN_BodyType_001(
                                                    md_EVCC_DC_MeteringReceiptRes_001(
                                                    oK,vc_DC_EVSEStatus))));
                    repeat;
                }
                else {
                    setverdict(v_vct, "Invalid signature of the MeteringReceiptReq " &
                                      "message was received.");
                }
            }
        }

        [] t_stopProcessing.timeout {
            setverdict(fail, "SUT did not react to EVSENotification 'renegotiate' " &
                             "within 'par_EVCC_StopCharging_Timeout'.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                       "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                    {currentDemandReq := ?},
                                                    {powerDeliveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                    "CurrentDemandReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(v_vct);
      }
      return getverdict;
    }

    function f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_007(in HAL_61851_Listener
                                                                v_HAL_61851_Listener)
                                                                runs on EVCC_Tester
                                                                return verdicttype {

        var boolean isStateB := false;
        var boolean isClosedTCP := false;
        var charstring v_charstring := "";

            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

        tc_EVCC_StateB_Shutdown_Timer.start(par_EVCC_StateB_Shutdown_Timeout);
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_StateB(pt_HAL_61851_Internal_Port){
               isStateB := true;
               tc_EVCC_StateB_Shutdown_Timer.stop;
               if(isClosedTCP == false) {repeat;}
            }
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.") {
               isClosedTCP := true;
               tc_CMN_TCP_Connection_Termination_Timer.stop;
               if((isStateB == false)) {repeat;}
            }
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout() {
               v_charstring := v_charstring & "SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'. ";
               if(tc_EVCC_StateB_Shutdown_Timer.running) {repeat;}
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        [] a_EVCC_StateB_Shutdown_Timeout() {
           v_charstring := v_charstring & "SUT has not changed to CP State B " &
                           "within 'par_EVCC_StateB_Shutdown_Timeout'. ";
           if(tc_CMN_TCP_Connection_Termination_Timer.running) {repeat;}
        }
    }
    if(not(isStateB and isClosedTCP) and v_charstring != "") {
       setverdict(fail,v_charstring);
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_008(HAL_61851_PwmMode_Type pwmMode)
                                                            runs on EVCC_Tester
                                                            return verdicttype  {
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                   md_CMN_BodyType_001(
                                   md_CMN_CMN_PowerDeliveryRes_001(
                                   oK, md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_CMN_DC_WeldingDetectionReq_001(?)))) {

               setverdict(fail,"Received WeldingDetectionReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_CMN_CMN_SessionStopReq_001(?)))) {

               setverdict(fail,"Received SessionStopReq message was not expected. " &
                               "CP State E/F should be detected before.");
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_009(in verdicttype v_vct)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

  var charstring v_message;
  var V2G_Message v_requestMessage;
  var integer v_loopCounter := 1;

 log("-------------------------------------------------------------");
 log("Number of received CurrentDemandReq messages: " & toString<integer>(v_loopCounter));
 log("-------------------------------------------------------------");
  if(not tc_V2G_SECC_Msg_Timer.running) {
      setverdict(inconc,"Msg performance time could not be fulfilled. " &
                        "The internal processing time should be optimized.");
  } else {tc_V2G_SECC_Msg_Timer.stop;}
  log("-------------------------------------------------------------");
  log("EVSE maximum charging limits:");
  log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumCurrentLimit)));
  log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumVoltageLimit)));
  log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumPowerLimit)));
  log("-------------------------------------------------------------");
  tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
  if(PICS_CMN_CMN_IdentificationMode == pnC) {
      pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                      md_CMN_MessageHeaderType_001(vc_SessionID),
                      md_CMN_BodyType_001(
                      md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                      vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                      false, false, false, vc_EVSEMaximumVoltageLimit,
                      vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                      vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, false))));
  } else {
     pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                     md_CMN_MessageHeaderType_001(vc_SessionID),
                     md_CMN_BodyType_001(
                     md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                     vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                     false, false, false, vc_EVSEMaximumVoltageLimit,
                     vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                     vc_EvseId, vc_SAScheduleTupleId, omit, false))));
  }
  alt {
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_PowerDeliveryReq_002(
                                    renegotiate, vc_SAScheduleTupleId, *,
                                    md_CMN_DC_EVPowerDeliveryParameter(
                                    md_CMN_DC_EVStatus_001(?,?,?),*,?)))))
                                    -> value v_requestMessage {

          tc_V2G_SECC_Sequence_Timer.stop;
          setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                          "be executed.");
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_CurrentDemandReq_001(
                                    md_CMN_DC_EVStatus_001(?,?,?),?,*,*,*,*,?,*,*,?))))
                                    -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        tc_V2G_SECC_Msg_Timer.start(
        par_V2G_SECC_Msg_CurrentDemand_Performance_Time);
        vc_EVRESSSOC := v_requestMessage.body.bodyElement.currentDemandReq.
        dC_EVStatus.eVRESSSOC;
        vc_EVTargetCurrent := v_requestMessage.body.bodyElement.currentDemandReq.
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        eVTargetCurrent;
f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                 cc_maximum_value_physicalValue_case1,
                                 v_vct);
vc_EVSEPresentCurrent := vc_EVTargetCurrent;
vc_EVTargetVoltage := v_requestMessage.body.bodyElement.currentDemandReq.
                        eVTargetVoltage;
f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                 cc_maximum_value_physicalValue_case2,
                                 v_vct);
if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
            eVMaximumCurrentLimit)) {
   vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                               currentDemandReq.eVMaximumCurrentLimit;
   f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                    vc_EVMaximumCurrentLimit, a,
                                    cc_maximum_value_physicalValue_case1,
                                    v_vct);
}
if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
            eVMaximumVoltageLimit)) {
   vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                               currentDemandReq.eVMaximumVoltageLimit;
   f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                    vc_EVMaximumVoltageLimit, v,
                                    cc_maximum_value_physicalValue_case2,
                                    v_vct);
}
if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
            eVMaximumPowerLimit)) {
   vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                             currentDemandReq.eVMaximumPowerLimit;
   f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                    vc_EVMaximumPowerLimit, w,
                                    cc_maximum_value_physicalValue_case4,
                                    v_vct);
}
log("------------------------------------------------------------");
log("EV maximum charging limits:");
log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumCurrentLimit)));
log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumVoltageLimit)));
if(vc_isPowLimit) {
    log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVMaximumPowerLimit)));
}
log("------------------------------------------------------------");
if(getverdict == pass) {
   setverdict(pass,"CurrentDemandReq message was correct.");
}

v_loopCounter := v_loopCounter + 1;
log("Current loopCounter: " & toString<integer>(v_loopCounter));
log("EVRESSSOC: " & toString<PercentValueType>(vc_EVRESSSOC) & "%.");
log("------------------------------------------------------------");

if(v_loopCounter == PICS_CMN_CMN_LoopCounter) {
    setverdict(inconc, "No Renegotiation was triggered by the SUT in the " &
                       "configured number of charging loop sequences" &
                       "(PICS_CMN_CMN_LoopCounter).");
}
if(not tc_V2G_SECC_Msg_Timer.running) {
    setverdict(inconc,"Msg performance time could not be fulfilled. " &
                      "The internal processing time should be optimized.");
} else {tc_V2G_SECC_Msg_Timer.stop;}
if(getverdict == pass) {
    log("------------------------------------------------------------");
    log("EVSE maximum charging limits:");
    log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
       vc_EVSEMaximumCurrentLimit)));
    log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
       vc_EVSEMaximumVoltageLimit)));
    log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
       vc_EVSEMaximumPowerLimit)));
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            log("------------------------------------------------------------");
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            if(PICS_CMN_CMN_IdentificationMode == pnC) {
                 pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                                vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                                false, false, false, vc_EVSEMaximumVoltageLimit,
                                vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                                vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, false))));
            } else {
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                               md_CMN_MessageHeaderType_001(vc_SessionID),
                               md_CMN_BodyType_001(
                               md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                               vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                               false, false, false, vc_EVSEMaximumVoltageLimit,
                               vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                               vc_EvseId, vc_SAScheduleTupleId, omit, false))));
            }
            repeat;
        }
    }
    [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                   "terminated by the SUT.");
    [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                {currentDemandReq := ?},
                                                {powerDeliveryReq := ?});
    [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                 "but cannot be decoded.");
    [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                                "CurrentDemandReq message, " &
                                                "but with invalid data content.");
    [] a_EVCC_Timeout(v_vct);
  }
  return getverdict;
}

function f_EVCC_DC_TB_VTB_ChargingLoopDcOrPowerDelivery_010(in verdicttype v_vct)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

  var charstring v_message;
  var V2G_Message v_requestMessage;
  var boolean v_isMetering := false;
  var boolean isSignatureOk := true;
  var integer v_loopCounter := 1;

  tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
  pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                              md_CMN_MessageHeaderType_001(vc_SessionID),
                              md_CMN_BodyType_001(
                              md_EVCC_DC_MeteringReceiptRes_001(
                              oK,vc_DC_EVSEStatus))));


  alt {
    [v_isMetering] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_PowerDeliveryReq_002(
                                    renegotiate, vc_SAScheduleTupleId, *,
                                    md_CMN_DC_EVPowerDeliveryParameter(
                                    md_CMN_DC_EVStatus_001(?,?,?),*,?)))))
                                    -> value v_requestMessage {

          tc_V2G_SECC_Sequence_Timer.stop;
          setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                          "be executed.");
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_CurrentDemandReq_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                            md_CMN_DC_EVStatus_001(?,?,?),?,*,*,*,*,?,*,*,?))))
                            -> value v_requestMessage {

tc_V2G_SECC_Sequence_Timer.stop;
tc_V2G_SECC_Msg_Timer.start(
par_V2G_SECC_Msg_CurrentDemand_Performance_Time);
vc_EVRESSSOC := v_requestMessage.body.bodyElement.currentDemandReq.
dC_EVStatus.eVRESSSOC;
vc_EVTargetCurrent := v_requestMessage.body.bodyElement.currentDemandReq.
                      eVTargetCurrent;
f_check_PhysicalValue_valueRange(cc_eVTargetCurrent, vc_EVTargetCurrent, a,
                                 cc_maximum_value_physicalValue_case1,
                                 v_vct);
vc_EVSEPresentCurrent := vc_EVTargetCurrent;
vc_EVTargetVoltage := v_requestMessage.body.bodyElement.currentDemandReq.
                      eVTargetVoltage;
f_check_PhysicalValue_valueRange(cc_eVTargetVoltage, vc_EVTargetVoltage, v,
                                 cc_maximum_value_physicalValue_case2,
                                 v_vct);

if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
            eVMaximumCurrentLimit)) {
   vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                               currentDemandReq.eVMaximumCurrentLimit;
   f_check_PhysicalValue_valueRange(cc_eVMaximumCurrentLimit,
                                    vc_EVMaximumCurrentLimit, a,
                                    cc_maximum_value_physicalValue_case1,
                                    v_vct);
}
if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
            eVMaximumVoltageLimit)) {
   vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                               currentDemandReq.eVMaximumVoltageLimit;
   f_check_PhysicalValue_valueRange(cc_eVMaximumVoltageLimit,
                                    vc_EVMaximumVoltageLimit, v,
                                    cc_maximum_value_physicalValue_case2,
                                    v_vct);
}
if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
            eVMaximumPowerLimit)) {
   vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                             currentDemandReq.eVMaximumPowerLimit;
   f_check_PhysicalValue_valueRange(cc_eVMaximumPowerLimit,
                                    vc_EVMaximumPowerLimit, w,
                                    cc_maximum_value_physicalValue_case4,
                                    v_vct);
}
log("-------------------------------------------------------------");
log("EV maximum charging limits:");
log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumCurrentLimit)));
log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumVoltageLimit)));
if(vc_isPowLimit) {
    log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVMaximumPowerLimit)));
}
log("-------------------------------------------------------------");
if(getverdict == pass) {
   setverdict(pass,"CurrentDemandReq message was correct.");
}

v_loopCounter := v_loopCounter + 1;
log("-------------------------------------------------------------");
log("Number of received CurrentDemandReq messages: " & toString<integer>(v_loopCounter));
log("EVRESSSOC: " & toString<PercentValueType>(vc_EVRESSSOC) & "%.");
log("-------------------------------------------------------------");

if(v_loopCounter == PICS_CMN_CMN_LoopCounter) {
    setverdict(inconc, "No Renegotiation was triggered by the SUT in the " &
                       "configured number of charging loop sequences" &
                       "(PICS_CMN_CMN_LoopCounter).");
}
if(not tc_V2G_SECC_Msg_Timer.running) {
    setverdict(inconc,"Msg performance time could not be fulfilled. " &
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        "The internal processing time should be optimized.");
     } else {tc_V2G_SECC_Msg_Timer.stop;}

     if(getverdict == pass) {
         log("------------------------------------------------------------");
         log("EVSE maximum charging limits:");
         log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
             vc_EVSEMaximumCurrentLimit)));
         log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
             vc_EVSEMaximumVoltageLimit)));
         log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
             vc_EVSEMaximumPowerLimit)));
         log("------------------------------------------------------------");
         tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
         pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                         md_CMN_MessageHeaderType_001(vc_SessionID),
                         md_CMN_BodyType_001(
                         md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                         vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                         false, false, false, vc_EVSEMaximumVoltageLimit,
                         vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                         vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, true))));
         repeat;
     }
 }
 [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_003(vc_SessionID,
                                md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_MeteringReceiptReq_001(
                                *, vc_SessionID, *, vc_MeterInfo))))
                                -> value v_requestMessage {

     tc_V2G_SECC_Sequence_Timer.stop;
     v_isMetering := true;
     var Signature authSignature := v_requestMessage.header.signature_;
     var BodyType meteringReceiptReqBodyType := v_requestMessage.body;

     f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                            cc_algorithm_method_2, cc_algorithm_method_3, v_vct);

     if(getverdict == pass) {

             isSignatureOk := f_verifySignature(meteringReceiptReqBodyType,
                                          authSignature,
                                          vc_contractSignatureCertChain.
                                          x509Certificate, v_vct);

         if(isSignatureOk) {
            setverdict(pass,"MeteringReceiptReq message was correct.");
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                   md_CMN_BodyType_001(
                                   md_EVCC_DC_MeteringReceiptRes_001(
                                   oK,vc_DC_EVSEStatus))));
            repeat;
         }
         else {
             setverdict(v_vct, "Invalid signature of the MeteringReceiptReq " &
                               "message was received.");
         }
     }
 }
 [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                        "terminated by the SUT.");
 [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                         {currentDemandReq := ?},
                                         {powerDeliveryReq := ?});
 [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                         "but cannot be decoded.");
 [] a_EVCC_Unexpected_Message_Content(v_vct, "Received PowerDeliveryReq or " &
                                         "CurrentDemandReq message, " &
                                         "but with invalid data content.");
 [] a_EVCC_Timeout(v_vct);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        }
        return getverdict;
    }

}
```

I.3.2.22 EVCC functions for WeldingDetection

```
module TestBehavior_EVCC_WeldingDetection {

    import from Timer_15118_2 all;
    import from Services_HAL_61851_1 all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from ComponentsAndPorts all;
    import from Pics_15118 all;
    import from Templates_CMN_WeldingDetection all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Templates_CMN_SessionStop all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_CMN_V2GMessage all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from TTlibrary_Logging all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from DataStructure_HAL_61851 all;
    import from Pixit_15118 all;
    import from LibFunctions_15118_2 all;
    import from Timer_15118 all;

    function f_EVCC_DC_TB_VTB_WeldingDetection_001(in HAL_61851_Listener v_HAL_61851_Listener,
                                                   in verdicttype v_vct)
                                                   runs on EVCC_Tester return verdicttype {

        var charstring v_message;
        var V2G_Message v_requestMessage;

            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_PowerDeliveryRes_001(
                                    oK, md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

             f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener,
                                        par_V2G_SECC_Sequence_Timeout, v_vct);

        alt {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                              md_CMN_MessageHeaderType_001(vc_SessionID),
                                              md_CMN_BodyType_001(
                                              md_CMN_DC_WeldingDetectionReq_001(?))))
                                              -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;

                    vc_EVSEPresentVoltage := { multiplier := 0, unit := v, value_ := 0  };
                setverdict(pass,"WeldingDetectionReq message was correct.");
            }
            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                           "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(v_vct, "Unexpected V2G message was received.",
                                            {weldingDetectionReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                         "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received WeldingDetectionReq message, " &
                                                        "but with invalid data content.");
            [] a_EVCC_Timeout(v_vct);
        }
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    return getverdict;
}

function f_EVCC_DC_TB_VTB_WeldingDetection_002(urn_iso_15118_2_2013_MsgDataTypes.
                                               ResponseCodeType responseCode)
                                               runs on EVCC_Tester return verdicttype {

    var charstring v_message;

      f_EVCC_setPwmMode(e_PosVolt12);

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_WeldingDetectionRes_001(
                                responseCode, vc_DC_EVSEStatus,
                                vc_EVSEPresentVoltage))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_WeldingDetection_003() runs on EVCC_Tester return verdicttype  {

    timer t_wait;

    // Wait until SUT message timeout
    t_wait.start(par_V2G_EVCC_Msg_Timeout_WeldingDetectionReq -
                 par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                              "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_WeldingDetection_004() runs on EVCC_Tester return verdicttype {

    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_CMN_SessionStop_Termination() {
            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);
            repeat;
        }
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_WeldingDetection_005(HAL_61851_PwmMode_Type pwmMode)
                                               runs on EVCC_Tester
                                               return verdicttype  {
    var boolean v_tcpTerminated := false;
    timer t_wait;

        f_EVCC_changeValidStateCondition(e_StateE,e_StateE);
        f_EVCC_setPwmMode(pwmMode);
        t_wait.start(par_CMN_waitForNextHAL);
        alt {
           [] t_wait.timeout;
           [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                         "terminated by the SUT " &
                                                         "before request message " &
                                                         "was sent by the Test System. " &
                                                         "The send operation will be " &
                                                         "canceled.") {
               v_tcpTerminated := true;
           }
        }
    if(not v_tcpTerminated) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     md_CMN_DC_WeldingDetectionRes_001(
                                     oK, vc_DC_EVSEStatus,
                                     vc_EVSEPresentVoltage))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                }
                [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                           "but TCP connection " &
                                                           "was >>NOT<< terminated by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                   setverdict(fail,"SUT has not terminated the TCP connection " &
                                   "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }
        }
        return getverdict;
    }

    function f_EVCC_DC_TB_VTB_WeldingDetection_006(in HAL_61851_Listener v_HAL_61851_Listener)
                                                 runs on EVCC_Tester return verdicttype {

        var charstring v_message;
        var V2G_Message v_requestMessage;
        var integer v_cnt := 0;

            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_CMN_PowerDeliveryRes_001(
                                    oK, md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

            f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener,
                                    par_V2G_SECC_Sequence_Timeout, fail);


        alt {
            [v_cnt == 0] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_WeldingDetectionReq_001(?))))
                                          -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_WeldingDetectionRes_001(
                                        oK,vc_DC_EVSEStatus,vc_EVSEPresentVoltage))));

                setverdict(pass,"WeldingDetectionReq message was correct.");
                v_cnt := v_cnt + 1;
                repeat;
            }
            [v_cnt == 1] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_WeldingDetectionReq_001(?))))
                                          -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass,"The repetition of WeldingDetectionReq message was correct.");
            }
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                          "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                        {weldingDetectionReq := ?});
            [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received WeldingDetectionReq message, " &
                                                       "but with invalid data content.");
            [] a_EVCC_Timeout(fail);
        }
        return getverdict;
    }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

I.3.2.23 EVCC functions for WeldingDetectionOrSessionStop

```
module TestBehavior_EVCC_WeldingDetectionOrSessionStop {

    import from Services_HAL_61851_1 all;
    import from ComponentsAndPorts all;
    import from TestBehavior_EVCC_PowerDelivery all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from TestBehavior_EVCC_CommonBehavior all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from Timer_15118 all;
    import from DataStructure_HAL_61851 all;
    import from Templates_CMN_SessionStop all;
    import from Templates_CMN_WeldingDetection all;
    import from Templates_CMN_PowerDelivery all;
    import from Templates_EVCC_PaymentServiceSelection all;
    import from Pics_15118_2 all;
    import from Pics_15118 all;
    import from Templates_CMN_V2GMessage all;
    import from DataStructure_V2G_TCP_TLS_Port_Control all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from TTlibrary_Logging all;
    import from Services_EVCC_PLCLinkStatus all;
    import from Pixit_15118 all;
    import from LibFunctions_15118_2 all;
    import from LibFunctions_15118_3 all;
    import from TestBehavior_EVCC_CmSetKey all;

    function f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_001(in HAL_61851_Listener
                                                                v_HAL_61851_Listener,
                                                                in verdicttype v_verdict1,
                                                                in verdicttype v_verdict2,
                                                                in verdicttype v_vct)
                                                                runs on EVCC_Tester
                                                                return verdicttype {

       var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_internalResponseMessage;
       var boolean isClosed := false;
       var charstring v_message;
       var V2G_Message v_requestMessage;
       var hexstring v_Nid;
       var hexstring v_Nmk;

           f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

       tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
       pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                   md_CMN_BodyType_001(
                                   md_CMN_CMN_PowerDeliveryRes_001(oK,
                                   md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

              f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener,
                                        par_V2G_SECC_Sequence_Timeout, v_vct);

       alt {
           [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                                md_CMN_BodyType_001(
                                                md_CMN_CMN_SessionStopReq_001(?))))
                                                -> value v_requestMessage {

              tc_V2G_SECC_Sequence_Timer.stop;
              vc_ChargingSession := v_requestMessage.body.bodyElement.
                                    sessionStopReq.chargingSession;
              setverdict(pass, "SessionStopReq message was correct.");

              if (vc_ChargingSession == terminate){
                 setverdict(v_verdict1, "V2G Communication Session has been finished.");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            else {
               setverdict(v_verdict2, "V2G Communication Session has been paused.");
            }

            // Explicitely send a SessionStopRes to EVCC
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_SessionStopRes_001(oK))));

            tc_CMN_TCP_Connection_Termination_Timer.start(
            par_CMN_TCP_Connection_Termination_Timeout);

            alt{
                [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                              "terminated by the SUT.");
                [] a_EVCC_Unexpected_Message_Content(v_vct, "Received any message, " &
                                                            "but TCP connection " &
                                                            "was >>NOT<< terminated " &
                                                            "by the SUT");
                [] a_EVCC_TCP_Connection_Termination_Timeout()  {
                        setverdict(v_vct,"SUT has not terminated the TCP connection " &
                                      "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
                }
            }

            f_EVCC_setPwmMode(e_PosVolt12);
            pt_HAL_61851_Port.clear;
            tc_CMN_ProcessingTime_Timer.start(par_T_match_leave + par_CMN_Transmission_Delay);

            if(getverdict == pass and vc_ChargingSession == terminate and
                PICS_CMN_CMN_CombinedTesting) {
                 if(vc_unmapSlac) {
                    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
                 }
                 f_EVCC_getPLCLinkTermination(par_T_match_leave +
                                         par_CMN_Transmission_Delay, fail);
                 if(vc_unmapSlac) {
                    unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
                 }
            } else if(getverdict == pass and vc_ChargingSession == pause) {
                 if(vc_unmapSlac) {
                    map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
                 }
                 // store Nid and Nmk
                 v_Nmk := vc_Nmk;
                 v_Nid := vc_Nid;
                 // generate new Nid and Nmk
                 vc_Nmk := f_randomHexStringGen(32);
                 vc_Nid := fx_generateNID(vc_Nmk);
                 f_EVCC_CMN_TB_VTB_CmSetKey_001();
                 if (getverdict == pass) {
                     f_EVCC_getPLCLinkTermination(par_T_match_leave, inconc);
                 }
                 if(vc_unmapSlac) {
                    unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
                 }
                 // set stored Nid and Nmk
                 vc_Nmk := v_Nmk;
                 vc_Nid := v_Nid;
            }
        }
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_WeldingDetectionReq_001(?))))
                                        -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
                vc_EVSEPresentVoltage := { multiplier := 0, unit := v, value_ := 0  };
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_DC_WeldingDetectionRes_001(
                                        oK,vc_DC_EVSEStatus,vc_EVSEPresentVoltage))));

            setverdict(pass, "WeldingDetectionReq message was correct. Waiting for " &
                                 "SessionStopReq message");
            repeat;
        }
        [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection was misleadingly " &
                                                  "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G message was received.",
                                                 {weldingDetectionReq := ?},
                                                 {sessionStopReq := ?});
        [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                 "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(v_vct, "Received SessionStopReq or " &
                                                  "WeldingDetectionReq message, but with " &
                                                  "invalid data content.");
        [] a_EVCC_Timeout(v_vct);
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_002(in HAL_61851_Listener
                                                            v_HAL_61851_Listener,
                                                            in verdicttype v_verdict1,
                                                            in verdicttype v_verdict2)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

    var V2G_TCP_TLS_Port_Control_Internal_MessageRes v_internalResponseMessage;
    var boolean isClosed := false;
    var charstring v_message;
    var boolean isWeldingDetection := false;
    var V2G_Message v_requestMessage;
    var hexstring v_Nid;
    var hexstring v_Nmk;

        f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_CMN_CMN_PowerDeliveryRes_001(
                                 oK, md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

        f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener, par_V2G_SECC_Sequence_Timeout, fail);

    alt
    {
      [isWeldingDetection == true]
      pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                       md_CMN_MessageHeaderType_001(vc_SessionID),
                                       md_CMN_BodyType_001(
                                       md_CMN_CMN_SessionStopReq_001(?))))
                                       -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            vc_ChargingSession := v_requestMessage.body.bodyElement.
                                  sessionStopReq.chargingSession;
            setverdict(pass, "SessionStopReq message was correct.");

            if (vc_ChargingSession == terminate){
               setverdict(v_verdict1, "V2G Communication Session has been finished.");
            }
            else {
               setverdict(v_verdict2, "V2G Communication Session has been paused.");
            }

            // Explicitely send a SessionStopRes to EVCC
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_CMN_CMN_SessionStopRes_001(oK))));
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt{
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }

        f_EVCC_setPwmMode(e_PosVolt12);
        pt_HAL_61851_Port.clear;
        tc_CMN_ProcessingTime_Timer.start(par_T_match_leave + par_CMN_Transmission_Delay);

        if(getverdict == pass and vc_ChargingSession == terminate and
            PICS_CMN_CMN_CombinedTesting) {
             if(vc_unmapSlac) {
                map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
             }
             f_EVCC_getPLCLinkTermination(par_T_match_leave +
                                          par_CMN_Transmission_Delay, fail);
             if(vc_unmapSlac) {
                unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
             }
        } else if(getverdict == pass and vc_ChargingSession == pause) {
             if(vc_unmapSlac) {
                map(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
             }
             // store Nid and Nmk
             v_Nmk := vc_Nmk;
             v_Nid := vc_Nid;
             // generate new Nid and Nmk
             vc_Nmk := f_randomHexStringGen(32);
             vc_Nid := fx_generateNID(vc_Nmk);
             f_EVCC_CMN_TB_VTB_CmSetKey_001();
             if (getverdict == pass) {
                 f_EVCC_getPLCLinkTermination(par_T_match_leave, inconc);
             }
             if(vc_unmapSlac) {
                unmap(mtc:pt_SLAC_Port, system:pt_SLAC_Port);
             }
             // set stored Nid and Nmk
             vc_Nmk := v_Nmk;
             vc_Nid := v_Nid;
        }
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_WeldingDetectionReq_001(?))))
                                        -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
            vc_EVSEPresentVoltage := { multiplier := 0, unit := v, value_ := 0  };
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_WeldingDetectionRes_001(
                                    oK,vc_DC_EVSEStatus,vc_EVSEPresentVoltage))));

        setverdict(pass, "WeldingDetectionReq message was correct. Waiting for " &
                         "SessionStopReq message");
        isWeldingDetection := true;
        repeat;
    }
    [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                  "terminated by the SUT.");
    [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(fail, "Unexpected V2G message was received.",
                                                {weldingDetectionReq := ?},
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                {sessionStopReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received SessionStopReq " &
                                                "or WeldingDetectionReq " &
                                                "message, but with invalid data content.");
        [] a_EVCC_Timeout(fail);
      }
      return getverdict;
   }

   function f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_003(in HAL_61851_Listener
                                                        v_HAL_61851_Listener,
                                                        in verdicttype v_vct)
                                                        runs on EVCC_Tester
                                                        return verdicttype {

         var charstring v_message;
         var V2G_Message v_requestMessage;

             f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

         tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
         pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_CMN_PowerDeliveryRes_001(
                                oK, md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

             f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener, par_V2G_SECC_Sequence_Timeout, v_vct);

         alt {
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_SessionStopReq_001(?)))) {

                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass, "SessionStopReq message was correct.");
            }
            [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_WeldingDetectionReq_001(?))))
                                          -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                    vc_EVSEPresentVoltage := { multiplier := 0, unit := v, value_ := 0  };
                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_WeldingDetectionRes_001(
                                    oK,vc_DC_EVSEStatus,vc_EVSEPresentVoltage))));

                setverdict(pass, "WeldingDetectionReq message was correct. Waiting for " &
                                  "SessionStopReq message");
                repeat;
            }
            [] a_EVCC_TCPConnection_Status_Listener(v_vct, "TCP or TLS connection " &
                                                        "was misleadingly " &
                                                        "terminated by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(v_vct, "Unexpected V2G " &
                                                               "message was received.",
                                                     {weldingDetectionReq := ?},
                                                     {sessionStopReq := ?});
            [isError()] a_EVCC_Unexpected_Message(v_vct, "Unknown message was received, " &
                                                     "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(v_vct, "Received SessionStopReq " &
                                                      "or WeldingDetectionReq" &
                                                     " message, but with invalid " &
                                                     "data content.");
            [] a_EVCC_Timeout(v_vct);
         }
         return getverdict;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   }

function f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_004(urn_iso_15118_2_2013_MsgDataTypes.
                                                            ResponseCodeType responseCode)
                                                            runs on EVCC_Tester
                                                            return verdicttype {

    var charstring v_message;

      f_EVCC_setPwmMode(e_PosVolt12);


    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                 md_CMN_BodyType_001(
                                 md_CMN_CMN_SessionStopRes_001(responseCode))));

    tc_CMN_TCP_Connection_Termination_Timer.start(
    par_CMN_TCP_Connection_Termination_Timeout);

    alt {
        [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                   "was >>NOT<< terminated by the SUT");
        [] a_EVCC_TCP_Connection_Termination_Timeout()  {
           setverdict(fail,"SUT has not terminated the TCP connection " &
                           "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_005() runs on EVCC_Tester
                                                              return verdicttype {

    timer t_wait;

    // Wait until SUT message timeout
    t_wait.start(par_V2G_EVCC_Msg_Timeout_SessionStopReq -
                 par_CMN_Transmission_Delay);
    pt_V2G_TCP_TLS_Port_Control.send(md_CMN_CMN_tcpTlsStatusReq_001(
                                 e_getPortStatus));

    alt {
        [] t_wait.timeout;
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                      "terminated by the SUT " &
                                                      "before timeout has expired.");
        [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive {
            setverdict(fail, "No V2G message was expected.");
        }
    }

    if(getverdict == pass) {

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, " &
                                                       "but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
    }
    return getverdict;
}

function f_EVCC_DC_TB_VTB_WeldingDetectionOrSessionStop_006() runs on EVCC_Tester
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                                    return verdicttype {

        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_PaymentServiceSelectionRes_001(oK))));

        tc_CMN_TCP_Connection_Termination_Timer.start(
        par_CMN_TCP_Connection_Termination_Timeout);

        alt {
            [] a_EVCC_TCPConnection_Status_Listener(pass, "TCP connection was " &
                                                          "terminated by the SUT.");
            [] a_EVCC_CMN_SessionStop_Termination() {
                tc_CMN_TCP_Connection_Termination_Timer.start(
                par_CMN_TCP_Connection_Termination_Timeout);
                repeat;
            }
            [] a_EVCC_Unexpected_Message_Content(fail, "Received any message, but TCP connection " &
                                                       "was >>NOT<< terminated by the SUT");
            [] a_EVCC_TCP_Connection_Termination_Timeout()  {
               setverdict(fail,"SUT has not terminated the TCP connection " &
                               "within 'par_CMN_TCP_Connection_Termination_Timeout'.");
            }
        }
        return getverdict;
    }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex J
(normative)

Function specifications for 15118-2 Reliability and Robustness Testing

J.1 General information

This annex covers all *test behaviors* for Reliability and Robustness in the ATS.

J.2 SECC functions

```
module TestBehavior_SECC_Reliability {

    import from Templates_CMN_Security all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from Templates_CMN_CableCheck all;
    import from Templates_CMN_PowerDelivery all;
    import from urn_iso_15118_2_2013_MsgDef language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from Services_HAL_61851_1 all;
    import from Pics_15118_2 all;
    import from Pixit_15118_2 all;
    import from Pics_15118 all;
    import from Timer_15118_2 all;
    import from Timer_15118_3 all;
    import from ComponentsAndPorts all;
    import from DataStructure_HAL_61851 all;
    import from TestBehavior_SECC_CommonBehavior all;
    import from LibFunctions_15118_2 all;
    import from TestBehavior_SECC_PowerDelivery all;
    import from TestBehavior_SECC_ChargeParameterDiscovery all;
    import from TestBehavior_SECC_ChargingStatus all;
    import from TestBehavior_SECC_MeteringReceipt all;
    import from TestBehavior_SECC_CableCheck all;
    import from TestBehavior_SECC_PreCharge all;
    import from TestBehavior_SECC_CurrentDemand all;
    import from TestBehavior_SECC_WeldingDetection all;
    import from TestBehavior_SECC_SessionStop all;
    import from Configurations_15118_2 all;
    import from Templates_CMN_V2GTcpTlsPortControl all;
    import from Templates_CMN_ChargingStatus all;
    import from Templates_CMN_V2GMessage all;
    import from Templates_CMN_CurrentDemand all;
    import from Services_Security all;

    import from Pixit_15118 all;
    import from Timer_15118 all;
    import from TTlibrary_Logging all;
    import from urn_iso_15118_2_2013_X509Def all;
    import from Templates_SECC_MeteringReceipt all;

    function f_SECC_AC_TB_VTB_Reliability_001(in HAL_61851_Listener v_HAL_61851_Listener,
                                              in float v_timeMin)
                                              runs on SECC_Tester return verdicttype {

          var boolean v_isChargingTime := false;
          timer t_chargingTime;
          var V2G_Message v_responseMessage;
          var integer v_loopCounter := 0;
          var charstring v_privateKey;
          var float v_timeForNextMessage := 1.0;

          while (vc_ChargeProgress != stop_ and getverdict == pass){
              // ChargeParameterDiscovery
              if (getverdict == pass) {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
      if(vc_ChargeProgress == start_) {
          if(PIXIT_SECC_CMN_SalesTariff == unknown) {
              f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
          } else {
              f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc);
          }
      }
}

// PowerDelivery
if (getverdict == pass) {
  if (vc_ChargeProgress == start_) {
      f_SECC_AC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
                                         inconc);
  }
}

t_chargingTime.start(v_timeMin * 60.0);
while(vc_EVSENotification == none_ and not v_isChargingTime and getverdict == pass) {
    // ChargingStatus
    if (getverdict == pass) {

      tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_ChargingStatusReq);
      pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(
                                         vc_SessionID),
                                         md_CMN_BodyType_001(
                                         m_CMN_AC_ChargingStatusReq_001)));
      alt{
         [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(
                                            vc_SessionID),
                                            md_CMN_BodyType_001 (
                                            md_CMN_AC_ChargingStatusRes_001 (
                                            oK, vc_EvseId,
                                            vc_SAScheduleTupleId,*,*,*,
                                            md_CMN_AC_EVSEStatus_001(?,?,?)))))
                                            -> value v_responseMessage {

          tc_V2G_EVCC_Msg_Timer.stop;
          if(getverdict == pass) {
                  setverdict(pass, "ChargingStatusRes message was correct.");
              vc_EVSENotification := v_responseMessage.body.bodyElement.
                                     chargingStatusRes.aC_EVSEStatus.eVSENotification;
              if(ispresent(v_responseMessage.body.bodyElement.chargingStatusRes.
                           meterInfo)) {
                  vc_MeterInfo := v_responseMessage.body.bodyElement.
                                  chargingStatusRes.meterInfo;
              }
          }
         }
         [] t_chargingTime.timeout {
            log("Defined charging time of " & toString<float>(v_timeMin) &
                  " min has reached. " &
                "Test system initiates normal shutdown.");
            vc_ChargeProgress := stop_;
            v_isChargingTime := true;
            repeat;
         }
         [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                    "was misleadingly " &
                                                   "terminated by the SUT.");
         [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                         {chargingStatusRes := ?});
         [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                  "but cannot be decoded.");
         [] a_SECC_Unexpected_Message_Content(fail, "Received ChargingStatusRes " &
                                                  "message, but with invalid " &
                                                  "data content.");
         [] a_SECC_Timeout(fail);
       }
    }
    v_loopCounter := v_loopCounter + 1;
        log("------------------------------------------------------------");
        log("Number of received ChargingStatusRes messages: " &
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                toString<integer>(v_loopCounter));
            log("------------------------------------------------------------");
        // MeteringReceipt
        if (getverdict == pass) {
          if(vc_receiptRequired and PICS_CMN_CMN_IdentificationMode == pnC){

              var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                                md_SECC_CMN_MeteringReceiptReq_001(
                                                "mReq", vc_SessionID,
                                                vc_MeterInfo, vc_SAScheduleTupleId));

              if(isbound(vc_privateKey)) {
                  v_privateKey := vc_privateKey;
              } else {
                  v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
              }

              var Signature aSignature := md_CMN_Signature_001(
                                          md_CMN_SignedInfoType_001("mReq",
                                          f_generateDigestFromBodyType(
                                          myMeteringReceiptBodyType)),
                                          v_privateKey);



              tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
              pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_003(
                                            vc_SessionID, aSignature),
                                            myMeteringReceiptBodyType));

              alt {
                  [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                                  md_CMN_MessageHeaderType_001(
                                                  vc_SessionID),
                                                  md_CMN_BodyType_001 (
                                                  md_SECC_AC_MeteringReceiptRes_001(oK,
                                                  md_CMN_AC_EVSEStatus_001(?,?,?)))))
                                                  -> value v_responseMessage {
                     tc_V2G_EVCC_Msg_Timer.stop;
                            setverdict(pass, "MeteringReceiptRes message was correct.");
                  }
                  [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                          "was misleadingly " &
                                                           "terminated by the SUT.");
                  [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message " &
                                                 "was received.",
                                                 {meteringReceiptRes := ?});
                  [isError()] a_SECC_Unexpected_Message(fail, "Unknown message " &
                                                        "was received, " &
                                                         "but cannot " &
                                                         "be decoded.");
                  [] a_SECC_Unexpected_Message_Content(fail, "Received " &
                                                       "MeteringReceiptRes " &
                                                       "message, but " &
                                                       "with invalid " &
                                                       "data content.");
                  [] a_SECC_Timeout(fail);
              }
          }
        }
        sleep(v_timeForNextMessage);
    }
    if (vc_EVSENotification == reNegotiation){
        vc_ChargeProgress := renegotiate;
        if(getverdict == pass) {
            f_SECC_AC_TB_VTB_PowerDelivery_001(renegotiate, v_HAL_61851_Listener,
                                               inconc);
        }
        if(getverdict == pass) {
           vc_EVMaxCurrent := { multiplier := 0, unit := a, value_ := 10 };
           if(PIXIT_SECC_CMN_SalesTariff == unknown) {
              f_SECC_AC_TB_VTB_ChargeParameterDiscovery_001(inconc);
            } else {
                f_SECC_AC_TB_VTB_ChargeParameterDiscovery_006(inconc);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    }
                    vc_ChargeProgress := start_;
                    vc_EVSENotification := none_;
                }
            }
        }
        return getverdict;
}

function f_SECC_DC_TB_VTB_Reliability_001(in HAL_61851_Listener v_HAL_61851_Listener,
                                          in float v_timeMin)
                                          runs on SECC_Tester
                                          return verdicttype {

  var boolean v_isChargingTime := false;
  var boolean v_isTargetCurrentZero := false;
  timer t_chargingTime;
  var V2G_Message v_responseMessage;
  var integer v_loopCounter := 0;
  var float v_timeForNextMessage := 1.0;
  var charstring v_privateKey;

  var PhysicalValueType v_EVMaximumCurrentLimitRestr := fx_setPhysicalValueType(
                                                         f_CMN_PhysicalValue_GetValue(
                                                         par_EVMaximumCurrentLimit)-1.0, "a");

  while (vc_ChargeProgress != stop_ and getverdict == pass){

      // ChargeParameterDiscovery
      if (getverdict == pass) {
          if (vc_ChargeProgress == start_) {
            if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
            } else {
                f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc);
            }
          }
      }

      // CableCheck
      if (getverdict == pass) {
        if(vc_ChargeProgress == start_) {
           f_SECC_DC_TB_VTB_CableCheck_001(v_HAL_61851_Listener, inconc);
        }
      }

      // PreCharge
      if (getverdict == pass) {
        if(vc_ChargeProgress == start_) {
          vc_EVTargetVoltage := par_EVBatteryVoltage;
          f_SECC_DC_TB_VTB_PreCharge_001(inconc);
        }
      }

      // PowerDelivery
      if (getverdict == pass) {
        if(vc_ChargeProgress == start_ ) {
            f_SECC_DC_TB_VTB_PowerDelivery_001(start_, v_HAL_61851_Listener,
                                               false, true, inconc);
        }
      }
      t_chargingTime.start(v_timeMin * 60.0);
      while(vc_EVSENotification == none_ and not v_isChargingTime and getverdict == pass) {

          // CurrentDemand
          if (getverdict == pass) {
                log("-------------------------------------------------------------");
                log("EV maximum charging limits:");
                log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                    vc_EVMaximumCurrentLimit)));
                log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                    vc_EVMaximumVoltageLimit)));
                log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                    vc_EVMaximumPowerLimit)));
                log("-------------------------------------------------------------");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
if(v_isTargetCurrentZero) {
  v_isChargingTime := true;
  vc_ChargeProgress := stop_;
}

tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_CurrentDemandReq);
pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                            md_CMN_MessageHeaderType_001(vc_SessionID),
                            md_CMN_BodyType_001(md_CMN_DC_CurrentDemandReq_001(
                            vc_DC_EVStatus,vc_EVTargetCurrent,
                            vc_EVMaximumVoltageLimit,
                            vc_EVMaximumCurrentLimit,vc_EVMaximumPowerLimit,
                            false, true ,
                            vc_remainingTimeToFullSoC,
                            vc_remainingTimeToBulkSoC,
                            vc_EVTargetVoltage))));

alt {
    [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     md_CMN_DC_CurrentDemandRes_001(
                                     oK,Templates_CMN_CableCheck.
                                     md_CMN_DC_EVSEStatus_001(?,?,*,?),
                                     ?,?,?,?,?,*,*,*,?,?,*,*))))
                                     -> value v_responseMessage {
        tc_V2G_EVCC_Msg_Timer.stop;
        vc_EVSEPresentVoltage  := v_responseMessage.body.bodyElement.
                                  currentDemandRes.eVSEPresentVoltage;
        vc_EVSEPresentCurrent  := v_responseMessage.body.bodyElement.
                                  currentDemandRes.eVSEPresentCurrent;
        if(ispresent(v_responseMessage.body.bodyElement.
                     currentDemandRes.eVSEMaximumCurrentLimit)) {
            vc_EVSEMaximumCurrentLimit := v_responseMessage.body.bodyElement.
                                          currentDemandRes.eVSEMaximumCurrentLimit;
        }
        if(ispresent(v_responseMessage.body.bodyElement.
                     currentDemandRes.eVSEMaximumVoltageLimit)) {
            vc_EVSEMaximumVoltageLimit := v_responseMessage.body.bodyElement.
                                            currentDemandRes.eVSEMaximumVoltageLimit;
        }
        if(ispresent(v_responseMessage.body.bodyElement.
                     currentDemandRes.eVSEMaximumPowerLimit)) {
            vc_EVSEMaximumPowerLimit := v_responseMessage.body.bodyElement.
                                        currentDemandRes.eVSEMaximumPowerLimit;
        }
        vc_EVSENotification := v_responseMessage.body.bodyElement.
                               currentDemandRes.dC_EVSEStatus.eVSENotification;
        vc_EvseId := v_responseMessage.body.bodyElement.currentDemandRes.eVSEID;
        if(ispresent(v_responseMessage.body.bodyElement.currentDemandRes.
                     meterInfo)) {
            vc_MeterInfo := v_responseMessage.body.bodyElement.
                            currentDemandRes.meterInfo;
        }
        if(ispresent(v_responseMessage.body.bodyElement.currentDemandRes.
                     dC_EVSEStatus.eVSEIsolationStatus)) {
            if(v_responseMessage.body.bodyElement.currentDemandRes.
               dC_EVSEStatus.eVSEIsolationStatus != valid) {
                    logWarn("The field 'eVSEIsolationStatus' should " &
                            "be set to 'valid'.");
            }
        }
        if(v_responseMessage.body.bodyElement.currentDemandRes.dC_EVSEStatus.
           eVSEStatusCode != eVSE_Ready) {
                logWarn("The field 'eVSEStatusCode' should be set " &
                        "to 'eVSE_Ready'.");
        }
            log("-------------------------------------------------------------");
            log("EVSE maximum charging limits:");
            log("EVSEMaximumCurrentLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
                vc_EVSEMaximumCurrentLimit)));
            log("EVSEMaximumVoltageLimit: " & toString<float>(
            f_CMN_PhysicalValue_GetValue(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    vc_EVSEMaximumVoltageLimit)));
                log("EVSEMaximumPowerLimit: " & toString<float>(
                f_CMN_PhysicalValue_GetValue(
                    vc_EVSEMaximumPowerLimit)));
                log("-------------------------------------------------------------");
            if(ispresent(v_responseMessage.body.bodyElement.
                         currentDemandRes.receiptRequired)) {
                vc_receiptRequired  := v_responseMessage.body.bodyElement.
                                       currentDemandRes.receiptRequired;
            }
            if(getverdict == pass) {

                    setverdict(pass,"CurrentDemandRes message was correct.");
            }
        }
        [] t_chargingTime.timeout {
           log("Defined charging time of " & toString<float>(v_timeMin) &
                  " min has reached. " &
               "Test system initiates normal shutdown.");
           vc_EVTargetCurrent.value_ := 0;
           v_isTargetCurrentZero := true;
           repeat;
        }
        [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                      "was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                         {currentDemandRes := ?});
        [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_SECC_Unexpected_Message_Content(fail, "Received CurrentDemandRes message, " &
                                                   "but with invalid data content.");
        [] a_SECC_Timeout(fail);
    }
}
v_loopCounter := v_loopCounter + 1;
    log("-------------------------------------------------------------");
    log("Number of received CurrentDemandRes messages: " &
         toString<integer>(v_loopCounter));
    log("-------------------------------------------------------------");
// MeteringReceipt
if (getverdict == pass) {
  if(vc_receiptRequired and PICS_CMN_CMN_IdentificationMode == pnC) {

      var BodyType myMeteringReceiptBodyType := md_CMN_BodyType_001(
                                                md_SECC_CMN_MeteringReceiptReq_001(
                                                "mReq", vc_SessionID,
                                                vc_MeterInfo, vc_SAScheduleTupleId));

      if(isbound(vc_privateKey)) {
          v_privateKey := vc_privateKey;
      } else {
          v_privateKey := fx_loadPrivateKey(cc_crt_contract_leaf_valid);
      }

      var Signature aSignature := md_CMN_Signature_001(
                                  md_CMN_SignedInfoType_001("mReq",
                                  f_generateDigestFromBodyType(
                                  myMeteringReceiptBodyType)),
                                  v_privateKey);

      tc_V2G_EVCC_Msg_Timer.start(par_V2G_EVCC_Msg_Timeout_MeteringReceiptReq);
      pt_V2G_TCP_TLS_ALM_SECC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_003(
                                    vc_SessionID, aSignature),
                                    myMeteringReceiptBodyType));

      alt {
          [] pt_V2G_TCP_TLS_ALM_SECC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(
                                            vc_SessionID),
                                            md_CMN_BodyType_001 (
                                            md_SECC_DC_MeteringReceiptRes_001(
                                            oK,Templates_CMN_PowerDelivery.
                                            md_CMN_DC_EVSEStatus_001(?,?,*,?)))))
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                    -> value v_responseMessage {
                                tc_V2G_EVCC_Msg_Timer.stop;
                                if(ispresent(v_responseMessage.body.bodyElement.
                                                meteringReceiptRes.eVSEStatus.
                                            dC_EVSEStatus.eVSEIsolationStatus)) {
                                    if(v_responseMessage.body.bodyElement.meteringReceiptRes.eVSEStatus.
                                       dC_EVSEStatus.eVSEIsolationStatus != valid) {
                                            logWarn("The field 'eVSEIsolationStatus' " &
                                                      "should be set to 'valid'.");
                                    }
                                }
                                if(v_responseMessage.body.bodyElement.meteringReceiptRes.eVSEStatus.
                                   dC_EVSEStatus.eVSEStatusCode != eVSE_Ready) {
                                         logWarn("The field 'eVSEStatusCode' should be " &
                                                   "set to 'eVSE_Ready'.");
                                }
                                       setverdict(pass, "MeteringReceiptRes message was correct.");
                            }
                            [] a_SECC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                                          "was misleadingly " &
                                                                          "terminated by the SUT.");
                            [] a_SECC_Unexpected_V2G_Message(fail, "Unexpected V2G message " &
                                                                   "was received.",
                                                        {meteringReceiptRes := ?});
                            [isError()] a_SECC_Unexpected_Message(fail, "Unknown message was " &
                                                                        "received, but cannot " &
                                                                        "be decoded.");
                            [] a_SECC_Unexpected_Message_Content(fail, "Received MeteringReceiptRes " &
                                                                       "message, but with invalid " &
                                                                       "data content.");
                            [] a_SECC_Timeout(fail);
                        }
                    }
                }
                sleep(v_timeForNextMessage);
            }
            if (vc_EVSENotification == reNegotiation) {
                vc_ChargeProgress := renegotiate;
                if (getverdict == pass) {
                  f_SECC_DC_TB_VTB_PowerDelivery_001(renegotiate, v_HAL_61851_Listener,
                                                     false, true, inconc);
                }
                if (getverdict == pass) {
                  vc_EVMaximumCurrentLimit := { multiplier := 0, unit := a, value_ := 50 };
                  if(PIXIT_SECC_CMN_SalesTariff == unknown) {
                      f_SECC_DC_TB_VTB_ChargeParameterDiscovery_001(inconc);
                  } else {
                      f_SECC_DC_TB_VTB_ChargeParameterDiscovery_006(inconc);
                  }
                  if(getverdict == pass) {
                      vc_EVTargetCurrent := { multiplier := 0, unit := a, value_ := 1 };
                      vc_ChargeProgress := start_;
                      vc_EVSENotification := none_;
                  }
                  }
            }

        }
        pt_V2G_TCP_TLS_ALM_SECC_Port.clear;
        return getverdict;
    }
}
```

J.3 EVCC functions

```
module TestBehavior_EVCC_Reliability {

    import from TestBehavior_EVCC_ChargingLoopDcOrPowerDelivery all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from XSDAUX language "XSD" all;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
import from Services_HAL_61851_1 all;
import from Timer_15118_2 all;
import from ComponentsAndPorts all;
import from Templates_CMN_ChargingStatus all;
import from Templates_CMN_PowerDelivery all;
import from Templates_EVCC_MeteringReceipt all;
import from Templates_CMN_CurrentDemand all;
import from Templates_CMN_PreCharge all;
import from Templates_CMN_WeldingDetection all;
import from Templates_CMN_V2GMessage all;
import from Templates_CMN_CableCheck all;
import from TestBehavior_EVCC_CommonBehavior all;
import from urn_iso_15118_2_2013_MsgDef all;
import from LibFunctions_15118_2 all;
import from TestBehavior_EVCC_PowerDelivery all;
import from Pics_15118 all;
import from Pics_15118_2 all;
import from TTlibrary_Logging all;
import from Templates_CMN_V2GTcpTlsPortControl all;
import from Templates_EVCC_ChargeParameterDiscovery all;
import from Services_Security all;
import from DataStructure_HAL_61851 all;

import from Pixit_15118 all;
import from Timer_15118 all;
import from urn_iso_15118_2_2013_MsgHeader all;
import from http_www_w3_org_2000_09_xmldsig all;
import from urn_iso_15118_2_2013_X509Def all;
import from Templates_CMN_Security all;
import from DataStructure_V2G all;
import from TestBehavior_EVCC_ChargeParameterDiscovery all;

function f_EVCC_AC_TB_VTB_Reliability_001(in HAL_61851_Listener v_HAL_61851_Listener,
                                          in float v_timeMin)
                                          runs on EVCC_Tester
                                          return verdicttype {

    var charstring v_message;
    var V2G_Message v_requestMessage;
    var integer v_renegotiationCounter := 0;
    var MessageHeaderType v_messageHeader;
    var charstring v_id;
    var SASchedules_group v_sASchedules;
    var boolean v_isFinished := true;
    var timer t_stopProcessing;
    timer t_chargingTime;
    var boolean v_isChargingTime := false;
    var integer v_loopCounter := 0;

    t_chargingTime.start(v_timeMin * 60.0);

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

    if(PICS_CMN_CMN_IdentificationMode == pnC) {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_AC_ChargingStatusRes_001(
                                    oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                    vc_MeterInfo, false, vc_AC_EVSEStatus))));

    } else {
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(
                                    vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_AC_ChargingStatusRes_001(
                                    oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                    omit, false, vc_AC_EVSEStatus))));
    }

    alt {
        [not vc_renegotiation] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                        md_CMN_V2G_Message_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    md_CMN_MessageHeaderType_001(
                                    vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_AC_PowerDeliveryReq_003(
                                    stop_, vc_SAScheduleTupleId, *)))){

  if(t_chargingTime.running) {
      pt_HAL_61851_Internal_Port.clear;
         setverdict(inconc, "PowerDeliveryReq message was not expected. " &
                         "The defined charging time was not considered by the SUT.");
  } else {
          timer stateTimer :=  par_V2G_SECC_CPState_Detection_Timeout +
                               par_CMN_Transmission_Delay;
          f_EVCC_confirmState(e_StateB, v_HAL_61851_Listener,stateTimer, fail, false);
      setverdict(pass,"PowerDeliveryReq message was correct.");

  }
  tc_V2G_SECC_Sequence_Timer.stop;
  vc_ChargeProgress := stop_;

}
[v_renegotiationCounter == 0]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_AC_PowerDeliveryReq_003(
                                renegotiate, vc_SAScheduleTupleId, *)))){

  tc_V2G_SECC_Sequence_Timer.stop;

  vc_renegotiation := true;
  v_renegotiationCounter := v_renegotiationCounter + 1;
  setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                 "be executed.");

  tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
  pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                               md_CMN_MessageHeaderType_001(vc_SessionID),
                               md_CMN_BodyType_001(
                               md_CMN_CMN_PowerDeliveryRes_001(
                               oK, md_CMN_AC_EVSEStatus_002(
                               vc_AC_EVSEStatus)))));
  vc_AC_EVSEStatus.eVSENotification := none_;
  repeat;

}
[vc_renegotiation and v_renegotiationCounter == 1]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_ChargeParamterDiscoveryReq_001(*,?,?))))
                                -> value v_requestMessage {

      v_isFinished := true;
      tc_V2G_SECC_Sequence_Timer.stop;
      if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                   maxEntriesSAScheduleTuple)) {
          vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                          chargeParameterDiscoveryReq.
                                          maxEntriesSAScheduleTuple;
      }
      v_renegotiationCounter := v_renegotiationCounter + 1;
      var EVChargeParameter_group evChargeParameter := v_requestMessage.body.
                                                       bodyElement.
                                                       chargeParameterDiscoveryReq.
                                                       eVChargeParameter;
      vc_EAmount := evChargeParameter.aC_EVChargeParameter.eAmount;
      vc_MeterInfo.meterReading := float2int(f_CMN_PhysicalValue_GetValue(vc_EAmount));
      vc_EVMaxVoltage := evChargeParameter.aC_EVChargeParameter.eVMaxVoltage;
      vc_EVMaxCurrent := evChargeParameter.aC_EVChargeParameter.eVMaxCurrent;
      vc_EVMinCurrent := evChargeParameter.aC_EVChargeParameter.eVMinCurrent;
      if(ispresent(evChargeParameter.aC_EVChargeParameter.departureTime)) {
          vc_DepartureTime := evChargeParameter.aC_EVChargeParameter.departureTime;
          if(vc_DepartureTime == 0) {
              logWarn("A departure time equals to 0s might lead to interoperability " &
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    "issues if the EVSE proceeds as defined in [V2G2-303].");
            }
        }
        if(getverdict == pass) {
          setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
          v_sASchedules := f_getSchedules(vc_SAScheduleTupleIdRenegotiation,
                                          vc_SAScheduleTupleIdRenegotiation2,
                                          vc_SAScheduleTupleIdRenegotiation3,
                                          vc_maxEntriesSAScheduleTuple);
          vc_SASchedule_List := v_sASchedules.sAScheduleList;

          if(PICS_CMN_CMN_IdentificationMode == pnC and vc_useSalesTariffs) {

              vc_myChargeParameterBodyType := md_CMN_BodyType_001(
                                        md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                        oK, finished, v_sASchedules,
                                        m_EVCC_AC_EVSEChargeParameter_001(
                                        vc_EVSENominalVoltage, vc_EVSEMaxCurrent)));

              vc_salesTariffIdDigestValueMap :=
                        fx_getIdDigestMapFromScheduleListType(
                        vc_SASchedule_List,
                        par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
                        - cc_salesTariff_processingTime_remains);

              if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
                  tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                  pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                        oK, ongoing, omit,
                                        m_EVCC_AC_EVSEChargeParameter_001(
                                        vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));
                  v_isFinished := false;
              } else {

                var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
                f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                                    vc_myChargeParameterBodyType,
                                                    vc_salesTariffIdDigestValueMap));
                var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                                 vc_privateKey);

                v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);
              }
            } else {
                v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
            }

            if(v_isFinished) {
                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        v_messageHeader,
                                        md_CMN_BodyType_001(
                                        md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                        oK, finished, v_sASchedules,
                                        m_EVCC_AC_EVSEChargeParameter_001(
                                        vc_EVSENominalVoltage,
                                        vc_EVSEMaxCurrent)))));
            }
            repeat;
        }
    }
    [vc_renegotiation and v_renegotiationCounter == 2]
    pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_AC_PowerDeliveryReq_003(
                                    start_, ?, ?)))) -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                            powerDeliveryReq.sAScheduleTupleID;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        vc_chargingProfile := v_requestMessage.body.bodyElement.
                              powerDeliveryReq.chargingProfile;

        vc_SAScheduleTupleId := v_SAScheduleTupleId;

        if(getverdict == pass) {
            f_checkValidMaxPowerEntries(vc_chargingProfile, vc_SASchedule_List,
                                        vc_SAScheduleTupleId, fail);
        }

        if(getverdict == pass) {
            vc_renegotiation := false;
            v_renegotiationCounter := 0;
            setverdict(pass,"PowerDeliveryReq message was correct.");
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(
                                        vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_PowerDeliveryRes_001(
                                        oK, md_CMN_AC_EVSEStatus_002(
                                        vc_AC_EVSEStatus)))));
            repeat;
        }
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      m_CMN_AC_ChargingStatusReq_001))) {

        tc_V2G_SECC_Sequence_Timer.stop;
        if(vc_AC_EVSEStatus.eVSENotification == stopCharging) {
            if(not t_stopProcessing.running) {
               t_stopProcessing.start(par_EVCC_StopCharging_Timeout);
            }
        }
        log("-------------------------------------------------------------");


        if(getverdict == pass) {
                setverdict(pass,"ChargingStatusReq message was correct. Waiting for " &
                                "PowerDeliveryReq message");
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            if(PICS_CMN_CMN_IdentificationMode == pnC) {
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_AC_ChargingStatusRes_001(
                                    oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                    vc_MeterInfo, false, vc_AC_EVSEStatus))));
            } else {
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_AC_ChargingStatusRes_001(
                                    oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                    omit, false, vc_AC_EVSEStatus))));
            }
            v_loopCounter := v_loopCounter + 1;
                log("-------------------------------------------------------------");
                log("Number of received ChargingStatusReq messages: " &
                    toString<integer>(v_loopCounter));
                log("-------------------------------------------------------------");
            repeat;

        }
    }
    [] t_chargingTime.timeout {
        log("Defined charging time of " & toString<float>(v_timeMin) & " min has reached. " &
            "Test system initiates normal shutdown.");
        v_isChargingTime := true;
        vc_AC_EVSEStatus.eVSENotification := stopCharging;
        vc_AC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);
        repeat;
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        [] t_stopProcessing.timeout {
            setverdict(fail, "SUT did not react to EVSENotification 'stopCharging' " &
                             "within 'par_EVCC_StopCharging_Timeout'.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(fail, "Unexpected V2G message was received.",
                                                   {chargingStatusReq := ?},
                                                   {powerDeliveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                   "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PowerDeliveryReq or " &
                                                   "ChargingStatusReq message, " &
                                                   "but with invalid data content.");
        [] a_EVCC_Timeout(fail);
    }

    return getverdict;
}

function f_EVCC_AC_TB_VTB_Reliability_002(in HAL_61851_Listener v_HAL_61851_Listener,
                                          in float v_timeMin)
                                          runs on EVCC_Tester
                                          return verdicttype {

    var charstring v_message;
    var V2G_Message v_requestMessage;
    var boolean v_isMetering := false;
    var integer v_renegotiationCounter := 0;
    var boolean isSignatureOk := true;
    var MessageHeaderType v_messageHeader;
    var charstring v_id;
    var SASchedules_group v_sASchedules;
    var boolean v_isFinished := true;
    var timer t_stopProcessing;
    timer t_chargingTime;
    var boolean v_isChargingTime := false;
    var integer v_loopCounter := 0;

    t_chargingTime.start(v_timeMin * 60.0);

    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);


    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(
                                      vc_SessionID),
                                      md_CMN_BodyType_001(
                                      md_CMN_AC_ChargingStatusRes_001(
                                      oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                                      vc_MeterInfo, true, vc_AC_EVSEStatus))));

    alt {
        [v_isMetering and not vc_renegotiation] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                          md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_AC_PowerDeliveryReq_003(
                                          stop_, vc_SAScheduleTupleId, *)))){

          if(t_chargingTime.running) {
              pt_HAL_61851_Internal_Port.clear;
              setverdict(inconc, "PowerDeliveryReq message was not expected. " &
                                 "The defined charging time was not considered by the SUT.");
          } else {
                  timer stateTimer :=  par_V2G_SECC_CPState_Detection_Timeout +
                                       par_CMN_Transmission_Delay;
                  f_EVCC_confirmState(e_StateB, v_HAL_61851_Listener,stateTimer, fail, false);
              setverdict(pass,"PowerDeliveryReq message was correct.");

          }

          tc_V2G_SECC_Sequence_Timer.stop;
          vc_ChargeProgress := stop_;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}
[v_isMetering and v_renegotiationCounter == 0]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_AC_PowerDeliveryReq_003(
                                renegotiate, vc_SAScheduleTupleId, *)))){

  tc_V2G_SECC_Sequence_Timer.stop;

  vc_renegotiation := true;
  v_renegotiationCounter := v_renegotiationCounter + 1;
  setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                 "be executed.");

  tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
  pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                               md_CMN_MessageHeaderType_001(vc_SessionID),
                               md_CMN_BodyType_001(
                               md_CMN_CMN_PowerDeliveryRes_001(
                               oK, md_CMN_AC_EVSEStatus_002(vc_AC_EVSEStatus)))));
  vc_AC_EVSEStatus.eVSENotification := none_;
  repeat;

}
[vc_renegotiation and v_renegotiationCounter == 1]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_ChargeParamterDiscoveryReq_001(*,?,?))))
                                -> value v_requestMessage {

      v_isFinished := true;
      tc_V2G_SECC_Sequence_Timer.stop;
      if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                   maxEntriesSAScheduleTuple)) {
          vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                          chargeParameterDiscoveryReq.
                                          maxEntriesSAScheduleTuple;
      }
      v_renegotiationCounter := v_renegotiationCounter + 1;
      var EVChargeParameter_group evChargeParameter := v_requestMessage.body.
                                                       bodyElement.
                                                       chargeParameterDiscoveryReq.
                                                       eVChargeParameter;
      vc_EAmount := evChargeParameter.aC_EVChargeParameter.eAmount;
      vc_MeterInfo.meterReading := float2int(f_CMN_PhysicalValue_GetValue(vc_EAmount));
      vc_EVMaxVoltage := evChargeParameter.aC_EVChargeParameter.eVMaxVoltage;
      vc_EVMaxCurrent := evChargeParameter.aC_EVChargeParameter.eVMaxCurrent;
      vc_EVMinCurrent := evChargeParameter.aC_EVChargeParameter.eVMinCurrent;
      if(ispresent(evChargeParameter.aC_EVChargeParameter.departureTime)) {
          vc_DepartureTime := evChargeParameter.aC_EVChargeParameter.departureTime;
          if(vc_DepartureTime == 0) {
            logWarn("A departure time equals to 0s might lead to interoperability " &
                    "issues if the EVSE proceeds as defined in [V2G2-303].");
          }
      }
      if(getverdict == pass) {
        setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
        v_sASchedules := f_getSchedules(vc_SAScheduleTupleIdRenegotiation,
                                        vc_SAScheduleTupleIdRenegotiation2,
                                        vc_SAScheduleTupleIdRenegotiation3,
                                        vc_maxEntriesSAScheduleTuple);
        vc_SASchedule_List := v_sASchedules.sAScheduleList;

        if(vc_useSalesTariffs) {

            vc_myChargeParameterBodyType := md_CMN_BodyType_001(
                                      md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                      oK, finished, v_sASchedules,
                                      m_EVCC_AC_EVSEChargeParameter_001(
                                      vc_EVSENominalVoltage, vc_EVSEMaxCurrent)));

            vc_salesTariffIdDigestValueMap :=
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        fx_getIdDigestMapFromScheduleListType(
                        vc_SASchedule_List,
                        par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
                        - cc_salesTariff_processingTime_remains);

                if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
                    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                          oK, ongoing, omit,
                                          m_EVCC_AC_EVSEChargeParameter_001(
                                          vc_EVSENominalVoltage, vc_EVSEMaxCurrent)))));
                    v_isFinished := false;
                } else {

                  var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
                  f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                                     vc_myChargeParameterBodyType,
                                                     vc_salesTariffIdDigestValueMap));
                  var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                                   vc_privateKey);

                  v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);
                }
              } else {
                  v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
              }

              if(v_isFinished) {
                  tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                  pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                v_messageHeader,
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                                oK, finished, v_sASchedules,
                                                m_EVCC_AC_EVSEChargeParameter_001(
                                                vc_EVSENominalVoltage,
                                                vc_EVSEMaxCurrent)))));
              }
              repeat;
          }
      }
      [vc_renegotiation and v_renegotiationCounter == 2]
      pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_AC_PowerDeliveryReq_003(
                                  start_, ?, ?)))) -> value v_requestMessage {

          tc_V2G_SECC_Sequence_Timer.stop;
          var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                              powerDeliveryReq.sAScheduleTupleID;

          vc_chargingProfile := v_requestMessage.body.bodyElement.
                                powerDeliveryReq.chargingProfile;

          vc_SAScheduleTupleId := v_SAScheduleTupleId;

          if(getverdict == pass) {
              vc_renegotiation := false;
              v_renegotiationCounter := 0;
              setverdict(pass,"PowerDeliveryReq message was correct.");
              tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
              pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(
                                          vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_CMN_PowerDeliveryRes_001(
                                          oK, md_CMN_AC_EVSEStatus_002(
                                          vc_AC_EVSEStatus)))));
              repeat;
          }
      }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
[] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    m_CMN_AC_ChargingStatusReq_001))) {
    tc_V2G_SECC_Sequence_Timer.stop;
    if(not v_isMetering) {
        setverdict(fail,"The SUT did not send a Metering Receipt message.");
    }
    if(vc_AC_EVSEStatus.eVSENotification == stopCharging) {
        if(not t_stopProcessing.running) {
            t_stopProcessing.start(par_EVCC_StopCharging_Timeout);
        }
    }


    if(getverdict == pass) {
            setverdict(pass,"ChargingStatusReq message was correct. Waiting for " &
                            "PowerDeliveryReq message");

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                              md_CMN_MessageHeaderType_001(vc_SessionID),
                              md_CMN_BodyType_001(
                              md_CMN_AC_ChargingStatusRes_001(
                              oK, vc_EvseId, vc_SAScheduleTupleId, omit,
                              vc_MeterInfo, true, vc_AC_EVSEStatus))));

        v_loopCounter := v_loopCounter + 1;
            log("-------------------------------------------------------------");
            log("Number of received ChargingStatusReq messages: " &
                toString<integer>(v_loopCounter));
            log("-------------------------------------------------------------");
        repeat;

    }
}
[] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_003(vc_SessionID,
                                    md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                    md_CMN_BodyType_001(
                                    md_EVCC_CMN_MeteringReceiptReq_001(
                                    *, vc_SessionID, *, vc_MeterInfo))))
                                    -> value v_requestMessage {

    tc_V2G_SECC_Sequence_Timer.stop;
    v_isMetering := true;
    var Signature authSignature := v_requestMessage.header.signature_;
    var BodyType meteringReceiptReqBodyType := v_requestMessage.body;

    f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                              cc_algorithm_method_2,
                              cc_algorithm_method_3, fail);

    if(getverdict == pass) {

            isSignatureOk := f_verifySignature(meteringReceiptReqBodyType,
                                               authSignature,
                                               vc_contractSignatureCertChain.
                                               x509Certificate, fail);

        if(isSignatureOk) {
                setverdict(pass,"MeteringReceiptReq message was correct.");
            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_AC_MeteringReceiptRes_001(
                                          oK,vc_AC_EVSEStatus))));
            repeat;
        }
        else {
            setverdict(fail, "Invalid signature of the MeteringReceiptReq message " &
                              "was received.");
        }
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
        [] t_chargingTime.timeout {
            log("Defined charging time of " & toString<float>(v_timeMin) &
                " min has reached. " &
                "Test system initiates normal shutdown.");
            v_isChargingTime := true;
            vc_AC_EVSEStatus.eVSENotification := stopCharging;
            vc_AC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
                f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);
            repeat;
        }
        [] t_stopProcessing.timeout {
            setverdict(fail, "SUT did not react to EVSENotification 'stopCharging' " &
                             "within 'par_EVCC_StopCharging_Timeout'.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection " &
                                                      "was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var3(fail, "Unexpected V2G message was received.",
                                                    {meteringReceiptReq := ?},
                                                    {chargingStatusReq := ?},
                                                    {powerDeliveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PowerDeliveryReq or " &
                                                    "ChargingStatusReq message, " &
                                                    "but with invalid data content.");
        [] a_EVCC_Timeout(fail);
    }

    return getverdict;
}

function f_EVCC_DC_TB_VTB_Reliability_001(in HAL_61851_Listener v_HAL_61851_Listener,
                                          in float v_timeMin)
                                          runs on EVCC_Tester
                                          return verdicttype {

  var charstring v_message;
  var V2G_Message v_requestMessage;
  var integer v_renegotiationCounter := 0;
  var boolean v_isCableCheck := false;
  var boolean v_isPreCharge := false;
  var MessageHeaderType v_messageHeader;
  var charstring v_id;
  var SASchedules_group v_sASchedules;
  var boolean v_isFinished := true;
  timer t1 := 0.1;
  timer t_cableCheck;
  timer t_keepVoltage := 3.0;
  var boolean v_isolationCheck := false;
  var boolean v_cableCheckFinished := false;
  var boolean v_stateNotDetected := true;
  var integer v_cnt := 0;
  var integer v_voltage_temp := 0;
  var integer v_cnt_PC := 0;
  timer t_stop;
  var timer t_stopProcessing;
  timer t_chargingTime;
  var boolean v_isChargingTime := false;
  var integer v_loopCounter := 0;

  t_chargingTime.start(v_timeMin * 60.0);

  var integer v_voltageStep := float2int(f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage))/
                                         cc_preChargeVoltageRampUpSteps;

  log("-------------------------------------------------------------");
  if(not tc_V2G_SECC_Msg_Timer.running) {
      setverdict(inconc,"Msg performance time could not be fulfilled. " &
                        "The internal processing time should be optimized.");
  } else {tc_V2G_SECC_Msg_Timer.stop;}
  log("-------------------------------------------------------------");
  log("EVSE maximum charging limits:");
  log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    vc_EVSEMaximumCurrentLimit)));
log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumVoltageLimit)));
log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumPowerLimit)));
log("-------------------------------------------------------------");

tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
if(PICS_CMN_CMN_IdentificationMode == pnC) {
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                        md_CMN_DC_CurrentDemandRes_001(
                        oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage,
                        vc_EVSEPresentCurrent, false, false, false,
                        vc_EVSEMaximumVoltageLimit, vc_EVSEMaximumCurrentLimit,
                        vc_EVSEMaximumPowerLimit, vc_EvseId, vc_SAScheduleTupleId,
                        vc_MeterInfo, false))));
} else {
    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                        md_CMN_DC_CurrentDemandRes_001(
                        oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage,
                        vc_EVSEPresentCurrent, false, false, false,
                        vc_EVSEMaximumVoltageLimit, vc_EVSEMaximumCurrentLimit,
                        vc_EVSEMaximumPowerLimit, vc_EvseId, vc_SAScheduleTupleId,
                        omit, false))));
}

alt {
  [not vc_renegotiation] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                      md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      mdw_CMN_DC_PowerDeliveryReq_003(
                                      stop_, vc_SAScheduleTupleId, *))))
                                      -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        if(t_chargingTime.running) {
            pt_HAL_61851_Internal_Port.clear;
            setverdict(inconc, "PowerDeliveryReq message was not expected. " &
                               "The defined charging time was not considered by the SUT.");
        } else {
               setverdict(pass, "PowerDeliveryReq message was correct.");
        }
  }
  [v_renegotiationCounter == 0]
  pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_PowerDeliveryReq_002(
                                    renegotiate, vc_SAScheduleTupleId, *,
                                    md_CMN_DC_EVPowerDeliveryParameter(
                                    md_CMN_DC_EVStatus_001(?,?,?),*,?))))) {

        tc_V2G_SECC_Sequence_Timer.stop;
        vc_renegotiation := true;
        v_renegotiationCounter := v_renegotiationCounter + 1;


        setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                        "be executed.");

            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                     md_CMN_MessageHeaderType_001(vc_SessionID),
                                     md_CMN_BodyType_001(
                                     md_CMN_CMN_PowerDeliveryRes_001(oK,
                                     md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

           f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener,
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    par_V2G_SECC_Sequence_Timeout, fail);
        vc_DC_EVSEStatus.eVSENotification := none_;
        repeat;
}
[vc_renegotiation and v_renegotiationCounter == 1]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                                *,?,?)))) -> value v_requestMessage {

      v_isFinished := true;
      vc_DC_EVSEStatus := {par_EVCC_NotificationMaxDelay, none_, invalid, eVSE_Ready};
      tc_V2G_SECC_Sequence_Timer.stop;
      if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
                   maxEntriesSAScheduleTuple)) {
          vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                          chargeParameterDiscoveryReq.
                                          maxEntriesSAScheduleTuple;
      }
      v_renegotiationCounter := v_renegotiationCounter + 1;
      var EVChargeParameter_group evChargeParameter := v_requestMessage.body.bodyElement.
                                                       chargeParameterDiscoveryReq.
                                                       eVChargeParameter;
      if(evChargeParameter.dC_EVChargeParameter.dC_EVStatus.eVReady == true) {
          logWarn("The field 'evReady' should be set to 'false'.");
      }
      vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                      chargeParameterDiscoveryReq.eVChargeParameter.
                      dC_EVChargeParameter.eVMaximumCurrentLimit;
      vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
      chargeParameterDiscoveryReq.eVChargeParameter.dC_EVChargeParameter.
      eVMaximumVoltageLimit;
      if(ispresent(v_requestMessage.body.bodyElement.
                   chargeParameterDiscoveryReq.eVChargeParameter.dC_EVChargeParameter.
                   eVMaximumPowerLimit)) {
            vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                      chargeParameterDiscoveryReq.eVChargeParameter.
                                      dC_EVChargeParameter.eVMaximumPowerLimit;
            vc_isPowLimit := true;
      }
      if(ispresent(evChargeParameter.dC_EVChargeParameter.departureTime)) {
          vc_DepartureTime := evChargeParameter.dC_EVChargeParameter.departureTime;
          if(vc_DepartureTime == 0) {
            logWarn("A departure time equals to 0s might lead to interoperability " &
                    "issues if the EVSE proceeds as defined in [V2G2-303].");
          }
      }
      log("-------------------------------------------------------------");
      log("EV maximum charging limits:");
      log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
          vc_EVMaximumCurrentLimit)));
      log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
          vc_EVMaximumVoltageLimit)));
      if(vc_isPowLimit) {
          log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
              vc_EVMaximumPowerLimit)));
      }
      log("-------------------------------------------------------------");
      if(getverdict == pass) {
        setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
        v_sASchedules := f_getSchedules(vc_SAScheduleTupleIdRenegotiation,
                                        vc_SAScheduleTupleIdRenegotiation2,
                                        vc_SAScheduleTupleIdRenegotiation3,
                                        vc_maxEntriesSAScheduleTuple);
        vc_SASchedule_List := v_sASchedules.sAScheduleList;
        log("-------------------------------------------------------------");
        log("EVSE maximum charging limits:");
        log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumCurrentLimit)));
        log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumVoltageLimit)));
        log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVSEMaximumPowerLimit)));
        log("EVSEMinimumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    vc_EVSEMinimumCurrentLimit)));
log("EVSEMinimumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVSEMinimumVoltageLimit)));
log("-------------------------------------------------------------");

if(PICS_CMN_CMN_IdentificationMode == pnC and vc_useSalesTariffs) {

    vc_myChargeParameterBodyType := md_CMN_BodyType_001(
                      md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                      oK, finished, v_sASchedules,
                      md_EVCC_DC_EVSEChargeParameter_002(
                      vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                      vc_EVSEMaximumPowerLimit,
                      vc_EVSEMaximumVoltageLimit,
                      vc_EVSEMinimumCurrentLimit,
                      vc_EVSEMinimumVoltageLimit,
                      vc_EVSECurrentRegulationTolerance,
                      vc_EVSEPeakCurrentRipple,
                      vc_EVSEEnergyToBeDelivered)));

    vc_salesTariffIdDigestValueMap :=
            fx_getIdDigestMapFromScheduleListType(
            vc_SASchedule_List,
            par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
            - cc_salesTariff_processingTime_remains);

    if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                              md_CMN_MessageHeaderType_001(vc_SessionID),
                              md_CMN_BodyType_001(
                              md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                              oK, ongoing, omit,
                              md_EVCC_DC_EVSEChargeParameter_002(
                              vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                              vc_EVSEMaximumPowerLimit,
                              vc_EVSEMaximumVoltageLimit,
                              vc_EVSEMinimumCurrentLimit,
                              vc_EVSEMinimumVoltageLimit,
                              omit,
                              vc_EVSEPeakCurrentRipple,
                              omit)))));
        v_isFinished := false;
    } else {

      var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
      f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                          vc_myChargeParameterBodyType,
                                          vc_salesTariffIdDigestValueMap));
      var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                       vc_privateKey);

      v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);
    }
  } else {
      v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
  }

  if(v_isFinished) {
        f_EVCC_changeValidStateCondition(e_StateB,vc_state);
      tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
      pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  v_messageHeader,
                                  md_CMN_BodyType_001(
                                  md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                  oK, finished, v_sASchedules,
                                  md_EVCC_DC_EVSEChargeParameter_002(
                                  vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                  vc_EVSEMaximumPowerLimit,
                                  vc_EVSEMaximumVoltageLimit,
                                  vc_EVSEMinimumCurrentLimit,
                                  vc_EVSEMinimumVoltageLimit,
                                  vc_EVSECurrentRegulationTolerance,
                                  vc_EVSEPeakCurrentRipple,
                                  vc_EVSEEnergyToBeDelivered)))));
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
            repeat;
        }
    }
    [vc_renegotiation and
    (v_renegotiationCounter == 2 or
     v_isCableCheck == true)] a_EVCC_StateCorD(
                                    pt_HAL_61851_Internal_Port, vc_state){
       v_stateNotDetected := false;
       repeat;
    }
    [vc_renegotiation and
    (v_renegotiationCounter == 2 or
     v_isCableCheck == true)]
     pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_CableCheckReq_001(
                                    md_CMN_DC_EVStatus_001(?,?,?)))))
                                    -> value v_requestMessage {

            tc_V2G_SECC_Sequence_Timer.stop;
            vc_DC_EVSEStatus.eVSEIsolationStatus := invalid;
            vc_DC_EVSEStatus.eVSEStatusCode := eVSE_IsolationMonitoringActive;
            v_cnt := v_cnt + 1;

            if(v_stateNotDetected){
                f_EVCC_StateChange_B2CorD(v_HAL_61851_Listener,
                                    par_EVCC_StateB_Shutdown_Timeout, fail);
                v_stateNotDetected := false;
            }

            if(v_cnt == 1) {
                t1.start;
            }

            if(v_renegotiationCounter == 2) {
               v_renegotiationCounter := v_renegotiationCounter + 1;
               v_isCableCheck := true;
            }
            tc_V2G_SECC_Sequence_Timer.stop;
            if(v_requestMessage.body.bodyElement.cableCheckReq.
               dC_EVStatus.eVReady == false) {
                logWarn("The field 'evReady' should be set to 'true'.");
            }


            setverdict(pass, "CableCheckReq message was correct.");

            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_CableCheckRes_001(
                                    oK, ongoing, vc_DC_EVSEStatus))));
            if(getverdict == pass) {
                repeat;
            }
    }
    [] t_cableCheck.timeout {
        v_cableCheckFinished := true;
        t1.start;
        repeat;
    }
    [] t1.timeout {
            if((not t_cableCheck.running) and (not v_cableCheckFinished)) {
                t_cableCheck.start(par_SECC_timeCableCheckVoltage);
            }
        if(not v_cableCheckFinished) {
            t1.start;
            repeat;

        } else {
            alt {
                [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(md_CMN_DC_CableCheckReq_001(
                                        md_CMN_DC_EVStatus_001(?,?,?)))))
                                        -> value v_requestMessage {

                    if(v_requestMessage.body.bodyElement.cableCheckReq.
                       dC_EVStatus.eVReady == false) {
                        logWarn("The field 'evReady' should be set to 'true'.");
                    }
                    tc_V2G_SECC_Sequence_Timer.stop;
                    setverdict(pass, "CableCheckReq message was correct.");
                }
                [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                          "misleadingly terminated " &
                                                          "by the SUT.");
                [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                          {cableCheckReq := ?});
                [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                         "but cannot be decoded.");
                [] a_EVCC_Unexpected_Message_Content(fail, "Received CableCheckReq message, " &
                                                        "but with invalid data content.");
                [] a_EVCC_Timeout(fail);
            }
            vc_DC_EVSEStatus.eVSEIsolationStatus := valid;
            vc_DC_EVSEStatus.eVSEStatusCode := eVSE_Ready;

            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_CableCheckRes_001(
                                          oK, finished, vc_DC_EVSEStatus))));
            repeat;
        }
    }
    [vc_renegotiation and
    (v_renegotiationCounter == 3 or
     v_isPreCharge == true)]
     pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_PreChargeReq_001(
                                    md_CMN_DC_EVStatus_001(?,?,?),?,?))))
                                    -> value v_requestMessage {

           tc_V2G_SECC_Sequence_Timer.stop;
           v_cnt_PC := v_cnt_PC + 1;
           vc_EVTargetVoltage := v_requestMessage.body.bodyElement.
                                 preChargeReq.eVTargetVoltage;
           if(v_cnt_PC <= cc_preChargeVoltageRampUpSteps) {
             v_voltage_temp := v_voltage_temp + v_voltageStep;
             vc_EVSEPresentVoltage.value_ := v_voltage_temp;
           }
           if(v_cnt_PC > cc_preChargeVoltageRampUpSteps) {
             vc_EVSEPresentVoltage.value_ := float2int(f_CMN_PhysicalValue_GetValue(
                                             vc_EVTargetVoltage));
           }
           if(v_renegotiationCounter == 3) {
              v_renegotiationCounter := v_renegotiationCounter + 1;
              v_isPreCharge := true;
           }
           if(v_requestMessage.body.bodyElement.preChargeReq.dC_EVStatus.
              eVReady == false) {
               logWarn("The field 'evReady' should be set to 'true'.");
           }
           vc_EVTargetCurrent := v_requestMessage.body.bodyElement.
                                 preChargeReq.eVTargetCurrent;
           vc_EVSEPresentCurrent := vc_EVTargetCurrent;


           if(getverdict == pass) {
              setverdict(pass, "PreChargeReq message was correct.");
           }

           tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_PreChargeRes_001(
                                    oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage))));
        if(getverdict == pass) {
          repeat;
        }
}
[vc_renegotiation and v_renegotiationCounter == 4]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_PowerDeliveryReq_002(
                                start_, ?, ?,
                                md_CMN_DC_EVPowerDeliveryParameter(
                                md_CMN_DC_EVStatus_001(?,?,?),*,false)))))
                                -> value v_requestMessage {

         tc_V2G_SECC_Sequence_Timer.stop;

         var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                             powerDeliveryReq.sAScheduleTupleID;

         vc_chargingProfile := v_requestMessage.body.bodyElement.
                               powerDeliveryReq.chargingProfile;

         vc_SAScheduleTupleId := v_SAScheduleTupleId;

         if(getverdict == pass) {
             vc_renegotiation := false;
             v_renegotiationCounter := 0;
             v_isCableCheck := false;
             v_isPreCharge := false;
             if(v_requestMessage.body.bodyElement.powerDeliveryReq.
                eVPowerDeliveryParameter.dC_EVPowerDeliveryParameter.
                dC_EVStatus.eVReady == false) {
                logWarn("The field 'evReady' should be set to 'true'.");
             }


             setverdict(pass, "PowerDeliveryReq message was correct.");

             tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
             pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                         md_CMN_MessageHeaderType_001(
                                         vc_SessionID),
                                         md_CMN_BodyType_001(
                                         md_CMN_CMN_PowerDeliveryRes_001(
                                         oK, md_CMN_DC_EVSEStatus_002(
                                         vc_DC_EVSEStatus)))));
             repeat;
         }
}
[] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_CurrentDemandReq_001(
                                md_CMN_DC_EVStatus_001(?,?,?),
                                ?,*,*,*,*,?,*,*,?))))
                                -> value v_requestMessage {

     tc_V2G_SECC_Sequence_Timer.stop;
     tc_V2G_SECC_Msg_Timer.start(
     par_V2G_SECC_Msg_CurrentDemand_Performance_Time);
     vc_EVRESSSOC := v_requestMessage.body.bodyElement.currentDemandReq.
     dC_EVStatus.eVRESSSOC;
     vc_EVTargetCurrent := v_requestMessage.body.bodyElement.
                           currentDemandReq.eVTargetCurrent;
     vc_EVSEPresentCurrent := vc_EVTargetCurrent;
     vc_EVTargetVoltage := v_requestMessage.body.bodyElement.
                           currentDemandReq.eVTargetVoltage;
     if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
                  eVMaximumCurrentLimit)) {
          vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                currentDemandReq.eVMaximumCurrentLimit;
}
if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
             eVMaximumVoltageLimit)) {
     vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                                currentDemandReq.eVMaximumVoltageLimit;
}
if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
             eVMaximumPowerLimit)) {
     vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                               currentDemandReq.eVMaximumPowerLimit;
}
    log("-------------------------------------------------------------");
    log("EV maximum charging limits:");
    log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVMaximumCurrentLimit)));
    log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVMaximumVoltageLimit)));
    if(vc_isPowLimit) {
        log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
            vc_EVMaximumPowerLimit)));
    }
    log("-------------------------------------------------------------");
if(v_requestMessage.body.bodyElement.currentDemandReq.
   dC_EVStatus.eVReady == false) {
         logWarn("The field 'evReady' should be set to 'true'.");
}
    log("EVRESSSOC: " & toString<PercentValueType>(vc_EVRESSSOC) & "%.");
    log("-------------------------------------------------------------");

if(vc_DC_EVSEStatus.eVSENotification == stopCharging) {
    if(not t_stopProcessing.running) {
        t_stopProcessing.start(par_EVCC_StopCharging_Timeout);
    }
}
if(not tc_V2G_SECC_Msg_Timer.running) {
    setverdict(inconc,"Msg performance time could not be fulfilled. " &
                      "The internal processing time should be optimized.");
} else {tc_V2G_SECC_Msg_Timer.stop;}
if(getverdict == pass) {
      setverdict(pass,"CurrentDemandReq message was correct. " &
                      "Waiting for PowerDeliveryReq message");

      log("-------------------------------------------------------------");
      log("EVSE maximum charging limits:");
      log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
         vc_EVSEMaximumCurrentLimit)));
      log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
         vc_EVSEMaximumVoltageLimit)));
      log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
         vc_EVSEMaximumPowerLimit)));
      log("-------------------------------------------------------------");
  tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
  // repeat sending current demand response message
  if(PICS_CMN_CMN_IdentificationMode == pnC) {
      pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                        md_CMN_MessageHeaderType_001(vc_SessionID),
                        md_CMN_BodyType_001(
                        md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                        vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                        false, false, false, vc_EVSEMaximumVoltageLimit,
                        vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                        vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, false))));
  } else {
     pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                       md_CMN_MessageHeaderType_001(vc_SessionID),
                       md_CMN_BodyType_001(
                       md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                       vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                       false, false, false, vc_EVSEMaximumVoltageLimit,
                       vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                       vc_EvseId, vc_SAScheduleTupleId, omit, false))));
  }
  v_loopCounter := v_loopCounter + 1;
       log("-------------------------------------------------------------");
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                log("Number of received CurrentDemandReq messages: " &
                    toString<integer>(v_loopCounter));
                log("------------------------------------------------------------");
            repeat;
          }
      }
      [] t_chargingTime.timeout {
          log("Defined charging time of " & toString<float>(v_timeMin) &
              " min has reached. " &
              "Test system initiates normal shutdown.");
          v_isChargingTime := true;
              vc_DC_EVSEStatus.eVSENotification := stopCharging;
              vc_DC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
          repeat;
      }
      [] t_stopProcessing.timeout {
          setverdict(fail, "SUT did not react to EVSENotification 'stopCharging' " &
                           "within 'par_EVCC_StopCharging_Timeout'.");
      }
      [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                    "terminated by the SUT.");
      [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var2(fail, "Unexpected V2G message was received.",
                                                  {currentDemandReq := ?},
                                                  {powerDeliveryReq := ?});
      [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                   "but cannot be decoded.");
      [] a_EVCC_Unexpected_Message_Content(fail, "Received PowerDeliveryReq or " &
                                                 "CurrentDemandReq message, " &
                                                 "but with invalid data content.");
      [] a_EVCC_Timeout(fail);
    }
    return getverdict;
  }

  function f_EVCC_DC_TB_VTB_Reliability_002(in HAL_61851_Listener v_HAL_61851_Listener,
                                            in float v_timeMin)
                                            runs on EVCC_Tester
                                            return verdicttype {

    var charstring v_message;
    var V2G_Message v_requestMessage;
    var integer v_renegotiationCounter := 0;
    var boolean v_isMetering := false;
    var boolean isSignatureOk := true;
    var boolean v_isCableCheck := false;
    var boolean v_isPreCharge := false;
    var MessageHeaderType v_messageHeader;
    var charstring v_id;
    var SASchedules_group v_sASchedules;
    var boolean v_isFinished := true;
    timer t1 := 0.1;
    timer t_cableCheck;
    timer t_keepVoltage := 3.0;
    var boolean v_isolationCheck := false;
    var boolean v_cableCheckFinished := false;
    var boolean v_stateNotDetected := true;
    var integer v_cnt := 0;
    var integer v_voltage_temp := 0;
    var integer v_cnt_PC := 0;
    timer t_stop;
    var timer t_stopProcessing;
    timer t_chargingTime;
    var boolean v_isChargingTime := false;
    var integer v_loopCounter := 0;

    t_chargingTime.start(v_timeMin * 60.0);

    var integer v_voltageStep := float2int(f_CMN_PhysicalValue_GetValue(vc_EVTargetVoltage))/
                                          cc_preChargeVoltageRampUpSteps;

    if(not tc_V2G_SECC_Msg_Timer.running) {
        setverdict(inconc,"Msg performance time could not be fulfilled. " &
                          "The internal processing time should be optimized.");
    } else {tc_V2G_SECC_Msg_Timer.stop;}
    log("------------------------------------------------------------");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
log("EVSE maximum charging limits:");
log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumCurrentLimit)));
log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumVoltageLimit)));
log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVSEMaximumPowerLimit)));
log("-------------------------------------------------------------");

tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);

pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                      md_CMN_MessageHeaderType_001(vc_SessionID),
                      md_CMN_BodyType_001(
                      md_CMN_DC_CurrentDemandRes_001(
                      oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage,
                      vc_EVSEPresentCurrent, false, false, false,
                      vc_EVSEMaximumVoltageLimit, vc_EVSEMaximumCurrentLimit,
                      vc_EVSEMaximumPowerLimit, vc_EvseId, vc_SAScheduleTupleId,
                      vc_MeterInfo, true))));

alt {
  [v_isMetering and not vc_renegotiation] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(
                                      md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_001(vc_SessionID),
                                      md_CMN_BodyType_001(
                                      mdw_CMN_DC_PowerDeliveryReq_003(
                                      stop_, vc_SAScheduleTupleId, *))))
                                      -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;
        if(t_chargingTime.running) {
            pt_HAL_61851_Internal_Port.clear;
            setverdict(inconc, "PowerDeliveryReq message was not expected. " &
                               "The defined charging time was not considered by the SUT.");
        } else {
            setverdict(pass, "PowerDeliveryReq message was correct.");
        }
  }
  [v_isMetering and v_renegotiationCounter == 0]
  pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_PowerDeliveryReq_002(
                                  renegotiate, vc_SAScheduleTupleId, *,
                                  md_CMN_DC_EVPowerDeliveryParameter(
                                  md_CMN_DC_EVStatus_001(?,?,?),*,?))))) {

        tc_V2G_SECC_Sequence_Timer.stop;
        vc_renegotiation := true;
        v_renegotiationCounter := v_renegotiationCounter + 1;


        setverdict(pass,"PowerDeliveryReq message was correct. Renegotiation shall " &
                        "be executed.");

            f_EVCC_changeValidStateCondition(e_StateValid_Charging,e_StateB);

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                   md_CMN_MessageHeaderType_001(vc_SessionID),
                                   md_CMN_BodyType_001(
                                   md_CMN_CMN_PowerDeliveryRes_001(oK,
                                   md_CMN_DC_EVSEStatus_002(vc_DC_EVSEStatus)))));

           f_EVCC_StateChange_CorD2B(v_HAL_61851_Listener,
                                  par_V2G_SECC_Sequence_Timeout, fail);
        vc_DC_EVSEStatus.eVSENotification := none_;
        repeat;
  }
  [vc_renegotiation and v_renegotiationCounter == 1]
  pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                          *,?,?)))) -> value v_requestMessage {

v_isFinished := true;
vc_DC_EVSEStatus := {par_EVCC_NotificationMaxDelay, none_, invalid, eVSE_Ready};
tc_V2G_SECC_Sequence_Timer.stop;
if(ispresent(v_requestMessage.body.bodyElement.chargeParameterDiscoveryReq.
             maxEntriesSAScheduleTuple)) {
    vc_maxEntriesSAScheduleTuple := v_requestMessage.body.bodyElement.
                                    chargeParameterDiscoveryReq.
                                    maxEntriesSAScheduleTuple;
}
v_renegotiationCounter := v_renegotiationCounter + 1;
var EVChargeParameter_group evChargeParameter := v_requestMessage.body.bodyElement.
                                                 chargeParameterDiscoveryReq.
                                                 eVChargeParameter;
if(evChargeParameter.dC_EVChargeParameter.dC_EVStatus.eVReady == true) {
    logWarn("The field 'evReady' should be set to 'false'.");
}
vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                chargeParameterDiscoveryReq.eVChargeParameter.
                dC_EVChargeParameter.eVMaximumCurrentLimit;
vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
chargeParameterDiscoveryReq.eVChargeParameter.dC_EVChargeParameter.
eVMaximumVoltageLimit;
if(ispresent(v_requestMessage.body.bodyElement.
             chargeParameterDiscoveryReq.eVChargeParameter.dC_EVChargeParameter.
             eVMaximumPowerLimit)) {
      vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                chargeParameterDiscoveryReq.eVChargeParameter.
                                dC_EVChargeParameter.eVMaximumPowerLimit;
      vc_isPowLimit := true;
}
if(ispresent(evChargeParameter.dC_EVChargeParameter.departureTime)) {
    vc_DepartureTime := evChargeParameter.dC_EVChargeParameter.departureTime;
    if(vc_DepartureTime == 0) {
      logWarn("A departure time equals to 0s might lead to interoperability " &
              "issues if the EVSE proceeds as defined in [V2G2-303].");
    }
}
log("-------------------------------------------------------------");
log("EV maximum charging limits:");
log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumCurrentLimit)));
log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
    vc_EVMaximumVoltageLimit)));
if(vc_isPowLimit) {
    log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
        vc_EVMaximumPowerLimit)));
}
log("-------------------------------------------------------------");

if(getverdict == pass) {
  setverdict(pass,"ChargeParamterDiscoveryReq message was correct.");
  v_sASchedules := f_getSchedules(vc_SAScheduleTupleIdRenegotiation,
                                  vc_SAScheduleTupleIdRenegotiation2,
                                  vc_SAScheduleTupleIdRenegotiation3,
                                  vc_maxEntriesSAScheduleTuple);
  vc_SASchedule_List := v_sASchedules.sAScheduleList;
  log("-------------------------------------------------------------");
  log("EVSE maximum charging limits:");
  log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumCurrentLimit)));
  log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumVoltageLimit)));
  log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMaximumPowerLimit)));
  log("EVSEMinimumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMinimumCurrentLimit)));
  log("EVSEMinimumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
      vc_EVSEMinimumVoltageLimit)));
  log("-------------------------------------------------------------");

  if(vc_useSalesTariffs) {

      vc_myChargeParameterBodyType := md_CMN_BodyType_001(
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                    oK, finished, v_sASchedules,
                                    md_EVCC_DC_EVSEChargeParameter_002(
                                    vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                    vc_EVSEMaximumPowerLimit,
                                    vc_EVSEMaximumVoltageLimit,
                                    vc_EVSEMinimumCurrentLimit,
                                    vc_EVSEMinimumVoltageLimit,
                                    vc_EVSECurrentRegulationTolerance,
                                    vc_EVSEPeakCurrentRipple,
                                    vc_EVSEEnergyToBeDelivered)));

                  vc_salesTariffIdDigestValueMap :=
                          fx_getIdDigestMapFromScheduleListType(
                          vc_SASchedule_List,
                          par_V2G_SECC_Msg_ChargeParameterDiscovery_Performance_Time
                          - cc_salesTariff_processingTime_remains);

                  if(not(ispresent(vc_salesTariffIdDigestValueMap))) {
                      tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                      pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                          oK, ongoing, omit,
                                          md_EVCC_DC_EVSEChargeParameter_002(
                                          vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                          vc_EVSEMaximumPowerLimit,
                                          vc_EVSEMaximumVoltageLimit,
                                          vc_EVSEMinimumCurrentLimit,
                                          vc_EVSEMinimumVoltageLimit,
                                          omit,
                                          vc_EVSEPeakCurrentRipple,
                                          omit)))));
                      v_isFinished := false;
                  } else {

                    var SignedInfo salesTariffSignedInfo := md_CMN_SignedInfoType_002(
                    f_calculateSalesTariffReferenceListFromChargeParameterDiscoveryRes(
                                                      vc_myChargeParameterBodyType,
                                                      vc_salesTariffIdDigestValueMap));
                    var Signature aSignature := md_CMN_Signature_001(salesTariffSignedInfo,
                                                                     vc_privateKey);

                    v_messageHeader := md_CMN_MessageHeaderType_003(vc_SessionID, aSignature);
                  }
                } else {
                    v_messageHeader := md_CMN_MessageHeaderType_001(vc_SessionID);
                }

                if(v_isFinished) {
                      f_EVCC_changeValidStateCondition(e_StateB,vc_state);
                    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                v_messageHeader,
                                                md_CMN_BodyType_001(
                                                md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                                                oK, finished, v_sASchedules,
                                                md_EVCC_DC_EVSEChargeParameter_002(
                                                vc_DC_EVSEStatus,vc_EVSEMaximumCurrentLimit,
                                                vc_EVSEMaximumPowerLimit,
                                                vc_EVSEMaximumVoltageLimit,
                                                vc_EVSEMinimumCurrentLimit,
                                                vc_EVSEMinimumVoltageLimit,
                                                vc_EVSECurrentRegulationTolerance,
                                                vc_EVSEPeakCurrentRipple,
                                                vc_EVSEEnergyToBeDelivered)))));
                }
                repeat;
            }
        }
        [vc_renegotiation and
        (v_renegotiationCounter == 2 or
         v_isCableCheck == true)] a_EVCC_StateCorD(
                                    pt_HAL_61851_Internal_Port, vc_state){
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            v_stateNotDetected := false;
            repeat;
        }
        [vc_renegotiation and
        (v_renegotiationCounter == 2 or
         v_isCableCheck == true)]
         pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_CableCheckReq_001(
                                        md_CMN_DC_EVStatus_001(?,?,?)))))
                                        -> value v_requestMessage {

                tc_V2G_SECC_Sequence_Timer.stop;
                vc_DC_EVSEStatus.eVSEIsolationStatus := invalid;
                vc_DC_EVSEStatus.eVSEStatusCode := eVSE_IsolationMonitoringActive;
                v_cnt := v_cnt + 1;

                if(v_stateNotDetected){
                    f_EVCC_StateChange_B2CorD(v_HAL_61851_Listener,
                                        par_EVCC_StateB_Shutdown_Timeout, fail);
                    v_stateNotDetected := false;
                }

                if(v_cnt == 1) {
                    t1.start;
                }

                if(v_renegotiationCounter == 2) {
                   v_renegotiationCounter := v_renegotiationCounter + 1;
                   v_isCableCheck := true;
                }
                tc_V2G_SECC_Sequence_Timer.stop;
                if(v_requestMessage.body.bodyElement.cableCheckReq.
                   dC_EVStatus.eVReady == false) {
                    logWarn("The field 'evReady' should be set to 'true'.");
                }


                setverdict(pass, "CableCheckReq message was correct.");

                tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_DC_CableCheckRes_001(
                                        oK, ongoing, vc_DC_EVSEStatus))));
                if(getverdict == pass) {
                    repeat;
                }
        }
        [] t_cableCheck.timeout {
            v_cableCheckFinished := true;
            t1.start;
            repeat;
        }
        [] t1.timeout {
                if((not t_cableCheck.running) and (not v_cableCheckFinished)) {
                    t_cableCheck.start(par_SECC_timeCableCheckVoltage);
                }
            if(not v_cableCheckFinished) {
                t1.start;
                repeat;

            } else {
                alt {
                    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                            md_CMN_MessageHeaderType_001(vc_SessionID),
                                            md_CMN_BodyType_001(md_CMN_DC_CableCheckReq_001(
                                            md_CMN_DC_EVStatus_001(?,?,?)))))
                                            -> value v_requestMessage {

                        if(v_requestMessage.body.bodyElement.cableCheckReq.
                           dC_EVStatus.eVReady == false) {
                            logWarn("The field 'evReady' should be set to 'true'.");
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                }
                tc_V2G_SECC_Sequence_Timer.stop;
                setverdict(pass, "CableCheckReq message was correct.");
            }
            [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP connection was " &
                                                          "misleadingly terminated " &
                                                          "by the SUT.");
            [] a_EVCC_Unexpected_V2G_Message(fail, "Unexpected V2G message was received.",
                                             {cableCheckReq := ?});
            [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                        "but cannot be decoded.");
            [] a_EVCC_Unexpected_Message_Content(fail, "Received CableCheckReq message, " &
                                                       "but with invalid data content.");
            [] a_EVCC_Timeout(fail);
        }
        vc_DC_EVSEStatus.eVSEIsolationStatus := valid;
        vc_DC_EVSEStatus.eVSEStatusCode := eVSE_Ready;

        tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
        pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                          md_CMN_MessageHeaderType_001(vc_SessionID),
                                          md_CMN_BodyType_001(
                                          md_CMN_DC_CableCheckRes_001(
                                          oK, finished, vc_DC_EVSEStatus))));
        repeat;
    }
}
[vc_renegotiation and
(v_renegotiationCounter == 3 or
 v_isPreCharge == true)]
 pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_PreChargeReq_001(
                                  md_CMN_DC_EVStatus_001(?,?,?),?,?))))
                                  -> value v_requestMessage {

       tc_V2G_SECC_Sequence_Timer.stop;
       v_cnt_PC := v_cnt_PC + 1;
       vc_EVTargetVoltage := v_requestMessage.body.bodyElement.
                             preChargeReq.eVTargetVoltage;
       if(v_cnt_PC <= cc_preChargeVoltageRampUpSteps) {
         v_voltage_temp := v_voltage_temp + v_voltageStep;
         vc_EVSEPresentVoltage.value_ := v_voltage_temp;
       }
       if(v_cnt_PC > cc_preChargeVoltageRampUpSteps) {
         vc_EVSEPresentVoltage.value_ := float2int(f_CMN_PhysicalValue_GetValue(
                                         vc_EVTargetVoltage));
       }
       if(v_renegotiationCounter == 3) {
          v_renegotiationCounter := v_renegotiationCounter + 1;
          v_isPreCharge := true;
       }
       if(v_requestMessage.body.bodyElement.preChargeReq.dC_EVStatus.
          eVReady == false) {
           logWarn("The field 'evReady' should be set to 'true'.");
       }
       vc_EVTargetCurrent := v_requestMessage.body.bodyElement.
                             preChargeReq.eVTargetCurrent;
       vc_EVSEPresentCurrent := vc_EVTargetCurrent;


       if(getverdict == pass) {
          setverdict(pass, "PreChargeReq message was correct.");
       }

       tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
       pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                  md_CMN_MessageHeaderType_001(vc_SessionID),
                                  md_CMN_BodyType_001(
                                  md_CMN_DC_PreChargeRes_001(
                                  oK, vc_DC_EVSEStatus, vc_EVSEPresentVoltage))));
       if(getverdict == pass) {
         repeat;
       }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}
[vc_renegotiation and v_renegotiationCounter == 4]
pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                md_CMN_MessageHeaderType_001(vc_SessionID),
                                md_CMN_BodyType_001(
                                md_CMN_DC_PowerDeliveryReq_002(
                                start_, ?, ?,
                                md_CMN_DC_EVPowerDeliveryParameter(
                                md_CMN_DC_EVStatus_001(?,?,?),*,false)))))
                                -> value v_requestMessage {

        tc_V2G_SECC_Sequence_Timer.stop;

        var SAIDType v_SAScheduleTupleId := v_requestMessage.body.bodyElement.
                                            powerDeliveryReq.sAScheduleTupleID;

        vc_chargingProfile := v_requestMessage.body.bodyElement.
                              powerDeliveryReq.chargingProfile;

        vc_SAScheduleTupleId := v_SAScheduleTupleId;

        if(getverdict == pass) {
            vc_renegotiation := false;
            v_renegotiationCounter := 0;
            v_isCableCheck := false;
            v_isPreCharge := false;
            if(v_requestMessage.body.bodyElement.powerDeliveryReq.
               eVPowerDeliveryParameter.dC_EVPowerDeliveryParameter.
               dC_EVStatus.eVReady == false) {
               logWarn("The field 'evReady' should be set to 'true'.");
            }


            setverdict(pass, "PowerDeliveryReq message was correct.");

            tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
            pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                        md_CMN_MessageHeaderType_001(
                                        vc_SessionID),
                                        md_CMN_BodyType_001(
                                        md_CMN_CMN_PowerDeliveryRes_001(
                                        oK, md_CMN_DC_EVSEStatus_002(
                                        vc_DC_EVSEStatus)))));
            repeat;
        }
}
[] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                    md_CMN_MessageHeaderType_001(vc_SessionID),
                                    md_CMN_BodyType_001(
                                    md_CMN_DC_CurrentDemandReq_001(
                                    md_CMN_DC_EVStatus_001(?,?,?),
                                    ?,*,*,*,*,?,*,*,?))))
                                    -> value v_requestMessage {

     tc_V2G_SECC_Sequence_Timer.stop;
     tc_V2G_SECC_Msg_Timer.start(
     par_V2G_SECC_Msg_CurrentDemand_Performance_Time);
     vc_EVRESSSOC := v_requestMessage.body.bodyElement.currentDemandReq.
     dC_EVStatus.eVRESSSOC;
     vc_EVTargetCurrent := v_requestMessage.body.bodyElement.
                           currentDemandReq.eVTargetCurrent;
     vc_EVSEPresentCurrent := vc_EVTargetCurrent;
     vc_EVTargetVoltage := v_requestMessage.body.bodyElement.
                           currentDemandReq.eVTargetVoltage;
     if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
                  eVMaximumCurrentLimit)) {
          vc_EVMaximumCurrentLimit := v_requestMessage.body.bodyElement.
                                      currentDemandReq.eVMaximumCurrentLimit;
     }
     if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
                  eVMaximumVoltageLimit)) {
          vc_EVMaximumVoltageLimit := v_requestMessage.body.bodyElement.
                                      currentDemandReq.eVMaximumVoltageLimit;
     }
     if(ispresent(v_requestMessage.body.bodyElement.currentDemandReq.
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    eVMaximumPowerLimit)) {
            vc_EVMaximumPowerLimit := v_requestMessage.body.bodyElement.
                                      currentDemandReq.eVMaximumPowerLimit;
        }
            log("-------------------------------------------------------------");
            log("EV maximum charging limits:");
            log("EVMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumCurrentLimit)));
            log("EVMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                vc_EVMaximumVoltageLimit)));
            if(vc_isPowLimit) {
                log("EVMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                    vc_EVMaximumPowerLimit)));
            }
            log("-------------------------------------------------------------");
        if(v_requestMessage.body.bodyElement.currentDemandReq.
           dC_EVStatus.eVReady == false) {
                logWarn("The field 'evReady' should be set to 'true'.");
        }
        if(not v_isMetering) {
            setverdict(fail,"The SUT did not send a Metering Receipt message.");
        }
             log("-------------------------------------------------------------");
            log("EVRESSSOC: " & toString<PercentValueType>(vc_EVRESSSOC) & "%.");
            log("-------------------------------------------------------------")


        if(vc_DC_EVSEStatus.eVSENotification == stopCharging) {
            if(not t_stopProcessing.running) {
                t_stopProcessing.start(par_EVCC_StopCharging_Timeout);
            }
        }
        if(not tc_V2G_SECC_Msg_Timer.running) {
            setverdict(inconc,"Msg performance time could not be fulfilled. " &
                              "The internal processing time should be optimized.");
        } else {tc_V2G_SECC_Msg_Timer.stop;}
        if(getverdict == pass) {
              setverdict(pass,"CurrentDemandReq message was correct. " &
                              "Waiting for PowerDeliveryReq message");
              log("-------------------------------------------------------------");
              log("EVSE maximum charging limits:");
              log("EVSEMaximumCurrentLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                 vc_EVSEMaximumCurrentLimit)));
              log("EVSEMaximumVoltageLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                 vc_EVSEMaximumVoltageLimit)));
              log("EVSEMaximumPowerLimit: " & toString<float>(f_CMN_PhysicalValue_GetValue(
                 vc_EVSEMaximumPowerLimit)));
              log("-------------------------------------------------------------");
          tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
          // repeat sending current demand response message
          pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                            md_CMN_MessageHeaderType_001(vc_SessionID),
                            md_CMN_BodyType_001(
                            md_CMN_DC_CurrentDemandRes_001(oK, vc_DC_EVSEStatus,
                            vc_EVSEPresentVoltage, vc_EVSEPresentCurrent,
                            false, false, false, vc_EVSEMaximumVoltageLimit,
                            vc_EVSEMaximumCurrentLimit, vc_EVSEMaximumPowerLimit,
                            vc_EvseId, vc_SAScheduleTupleId, vc_MeterInfo, true))));

          v_loopCounter := v_loopCounter + 1;
               log("-------------------------------------------------------------");
               log("Number of received CurrentDemandReq messages: " &
                   toString<integer>(v_loopCounter));
               log("-------------------------------------------------------------");
          repeat;
        }
    }
    [] pt_V2G_TCP_TLS_ALM_EVCC_Port.receive(md_CMN_V2G_Message_001(
                                      md_CMN_MessageHeaderType_003(vc_SessionID,
                                      md_CMN_Signature_002(mw_CMN_SignedInfoType_003)),
                                      md_CMN_BodyType_001(
                                      md_EVCC_CMN_MeteringReceiptReq_001(
                                      *, vc_SessionID, *, vc_MeterInfo))))
                                      -> value v_requestMessage{
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            tc_V2G_SECC_Sequence_Timer.stop;
            v_isMetering := true;
            var Signature authSignature := v_requestMessage.header.signature_;
            var BodyType meteringReceiptReqBodyType := v_requestMessage.body;

            f_checkSignatureConstFields(authSignature, cc_algorithm_method_1,
                                        cc_algorithm_method_2, cc_algorithm_method_3, fail);

            if(getverdict == pass) {

                    isSignatureOk := f_verifySignature(meteringReceiptReqBodyType,
                                                       authSignature,
                                                       vc_contractSignatureCertChain.
                                                       x509Certificate, fail);

                if(isSignatureOk) {

                        setverdict(pass,"MeteringReceiptReq message was correct.");
                    tc_V2G_SECC_Sequence_Timer.start(par_V2G_SECC_Sequence_Timeout);
                    pt_V2G_TCP_TLS_ALM_EVCC_Port.send(md_CMN_V2G_Message_001(
                                                 md_CMN_MessageHeaderType_001(vc_SessionID),
                                                 md_CMN_BodyType_001(
                                                 md_EVCC_DC_MeteringReceiptRes_001(
                                                 oK,vc_DC_EVSEStatus))));
                    repeat;
                }
                else {
                    setverdict(fail, "Invalid signature of the MeteringReceiptReq " &
                                     "message was received.");
                }
            }
        }
        [] t_chargingTime.timeout {
            log("Defined charging time of " & toString<float>(v_timeMin) &
                " min has reached. " &
                "Test system initiates normal shutdown.");
            v_isChargingTime := true;
                vc_DC_EVSEStatus.eVSENotification := stopCharging;
                vc_DC_EVSEStatus.notificationMaxDelay := par_EVCC_NotificationMaxDelay;
            repeat;
        }

        [] t_stopProcessing.timeout {
            setverdict(fail, "SUT did not react to EVSENotification 'stopCharging' " &
                             "within 'par_EVCC_StopCharging_Timeout'.");
        }
        [] a_EVCC_TCPConnection_Status_Listener(fail, "TCP or TLS connection was misleadingly " &
                                                      "terminated by the SUT.");
        [] a_EVCC_Unexpected_V2G_Message_PIXIT_Var3(fail, "Unexpected V2G message was received.",
                                                    {meteringReceiptReq := ?},
                                                    {currentDemandReq := ?},
                                                    {powerDeliveryReq := ?});
        [isError()] a_EVCC_Unexpected_Message(fail, "Unknown message was received, " &
                                                    "but cannot be decoded.");
        [] a_EVCC_Unexpected_Message_Content(fail, "Received PowerDeliveryReq or " &
                                                   "CurrentDemandReq message, " &
                                                   "but with invalid data content.");
        [] a_EVCC_Timeout(fail);
      }
      return getverdict;
    }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex K
(normative)

Template specifications for V2G TCP/TLS Port Control

K.1 General information

This annex covers all V2G TCP/TLS Port Control message *templates* in the ATS independent on the SUT type.

K.2 V2G TCP/TLS port control templates

```
module Templates_CMN_V2GTcpTlsPortControl {

    import from DataStructure_V2G_TCP_TLS_Port_Control language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_X509Def language "XSD" all;

    template V2G_TCP_TLS_Port_Control_Internal_MessageReq md_CMN_CMN_tcpTlsStatusReq_001(
                                                template V2G_TCP_TLS_Port_Control_Command_TYPE
                                                p_command) := {
        command := p_command
    };

    template V2G_TCP_TLS_Port_Control_Internal_MessageRes md_CMN_CMN_tcpTlsStatusRes_001(
                                template  V2G_TCP_TLS_Port_Control_Command_TYPE p_command,
                                template V2G_TCP_TLS_Port_Control_Ack_TYPE p_ack,
                                template boolean p_status,
                                template charstring p_reason) := {
        command := p_command,
        ack := p_ack,
        result := {portStatus := {p_status, p_reason}}
    }

    template V2G_TCP_TLS_Port_Control_Internal_MessageRes md_CMN_CMN_tcpTlsStatusRes_002(
                                            template V2G_TCP_TLS_Port_Control_Command_TYPE
                                            p_command,
                                            template V2G_TCP_TLS_Port_Control_Ack_TYPE p_ack,
                                            template integer p_portNumber) := {
        command := p_command,
        ack := p_ack,
        result := {
            portNumber := {p_portNumber}
        }
    };



    template V2G_TCP_TLS_Parameter md_CMN_CMN_TcpParameter_001(
                                template(present) hexstring p_seccIpaddress,
                                template(present) integer p_seccPort,
                                template hexstring p_security) := {

        seccIpaddress := p_seccIpaddress,
        seccPort := p_seccPort,
        security := p_security
    };

    template V2G_TCP_TLS_Parameter md_CMN_CMN_TlsParameter_001(
                                template(present) hexstring p_seccIpaddress,
                                template(present) integer p_seccPort,
                                template hexstring p_security,
                                template(present) charstring p_cipherSuite,
                                template charstring p_enabledProtocol,
                                template CertificateChainType p_certChain) := {

        seccIpaddress := p_seccIpaddress,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        seccPort := p_seccPort,
        security := p_security,
        tlsParameter := {
                cipherSuite := p_cipherSuite,
                enabledProtocol := p_enabledProtocol,
                result := {
                    certificateChain := p_certChain
                }
        }

    };

    template V2G_TCP_TLS_Parameter md_CMN_CMN_TlsParameter_002(
                                   template(present) hexstring p_seccIpaddress,
                                   template(present) integer p_seccPort,
                                   template hexstring p_security,
                                   template(present) charstring p_cipherSuite,
                                   template charstring p_enabledProtocol,
                                   template X509CertificateType p_certificate) := {

        seccIpaddress := p_seccIpaddress,
        seccPort := p_seccPort,
        security := p_security,
        tlsParameter := {
                cipherSuite := p_cipherSuite,
                enabledProtocol := p_enabledProtocol,
                result := {
                    certificate := p_certificate
                }
        }

    };
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex L
(normative)

Template specifications for 15118-2 V2GTP

L.1 General information

This annex covers all V2GTP *templates* in the ATS.

L.2 Common templates

```
module Templates_CMN_V2GTPMessage {

    import from DataStructure_V2GTP all;
    import from DataStructure_SDP all;
    import from urn_iso_15118_2_2013_MsgDef all;
    import from urn_iso_15118_2_2010_AppProtocol all;

    template V2GTP_Message md_CMN_V2GTP_Message_001(
                           template (present) V2GTP_header_TYPE p_v2gtp_header,
                           template (present) V2GTP_payload_TYPE p_v2gtp_payload) := {

         v2gtp_header := p_v2gtp_header,
         v2gtp_payload := p_v2gtp_payload

    }

    template V2GTP_payload_TYPE md_CMN_V2GTP_payload_TYPE_001(
                                  template (present) SDP_Message p_sdpMessage) := {

         content:={
             sdp_message:= p_sdpMessage
         }
     }

    template V2GTP_payload_TYPE md_CMN_V2GTP_payload_TYPE_002(
                               template (present) V2G_Message p_v2g_message) := {

         content:={
             v2g_message:= p_v2g_message
         }
     }

    template V2GTP_payload_TYPE md_CMN_V2GTP_payload_TYPE_003(
                               template (present) SupportedAppProtocolReq
                               p_supportedAppProtocolReq_message) := {
         content:={
            supportedAppProtocolReq_message:= p_supportedAppProtocolReq_message
         }
     }

    template V2GTP_payload_TYPE md_CMN_V2GTP_payload_TYPE_004(
                               template (present) SupportedAppProtocolRes
                               p_supportedAppProtocolRes_message) := {
         content:={
            supportedAppProtocolRes_message:= p_supportedAppProtocolRes_message
         }
     }

    template V2GTP_header_TYPE md_CMN_V2GTP_header_TYPE_001(
                              template (present) ProtocolVersion_TYPE p_protocolVersion,
                              template (present) InvProtocolVersion_TYPE p_invProtocolVersion,
                              template (present) PayloadType_TYPE p_payloadType,
                              template PayloadLength_TYPE p_payloadLength) := {

        protocolVersion := p_protocolVersion,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            invProtocolVersion := p_invProtocolVersion,
            payloadType := p_payloadType,
            payloadLength := p_payloadLength

        }
    }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex M
(normative)

Template specifications for 15118-2 SDP messages

M.1 General information

This annex covers all SDP *templates* in the ATS.

M.2 Common templates

```
module Templates_CMN_SDP {

    import from DataStructure_SDP all;

    template SDP_Message md_CMN_SDP_Request_TYPE_001(
                         template (present) Security_TYPE p_security,
                         template (present) TransportProtocol_TYPE p_transportProtocol) := {
         content:= {
             sdp_Request:= {
                 security := p_security,
                 transportProtocol := p_transportProtocol
             }
         }
    }

    template SDP_Message md_CMN_SDP_Response_TYPE_001(
                         template (present) SECC_IPaddress_TYPE p_secc_IPaddress,
                         template (present) SECC_Port_TYPE p_secc_Port,
                         template (present) Security_TYPE p_security,
                         template (present) TransportProtocol_TYPE p_transportProtocol) := {
         content:= {
             sdp_Response:= {
                 secc_IPaddress := p_secc_IPaddress,
                 secc_Port := p_secc_Port,
                 security := p_security,
                 transportProtocol := p_transportProtocol
             }
         }
     }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex N
(normative)

Template specifications for 15118-2 V2G application layer messages

N.1 General information

This annex covers all V2G application layer message *templates* in the ATS.

N.2 Common templates

```
module Templates_CMN_V2GMessage {

    import from urn_iso_15118_2_2013_MsgDef all;
    import from urn_iso_15118_2_2013_MsgBody all;
    import from urn_iso_15118_2_2013_MsgHeader all;
    import from http_www_w3_org_2000_09_xmldsig all;
    import from XSDAUX all;

    template V2G_Message md_CMN_V2G_Message_001(
                        template (present) MessageHeaderType p_Header,
                        template (present) BodyType p_Body) := {

        header := p_Header, body := p_Body
    }
    with {
        encode "ISO_V2G"
        }

    template BodyType md_CMN_BodyType_001(
                      template BodyElement_group p_BodyElement) := {

        bodyElement := p_BodyElement
    }

    template MessageHeaderType md_CMN_MessageHeaderType_001(
                               template (present) octetstring p_SessionID) := {

        sessionID := p_SessionID,
        notification := omit,
        signature_ := omit
    }

    template MessageHeaderType md_CMN_MessageHeaderType_002(
                               template (present) octetstring p_SessionID) := {

        sessionID := p_SessionID,
        notification := omit,
        signature_ := *
    }

    template MessageHeaderType md_CMN_MessageHeaderType_003(
                               template (present) octetstring p_SessionID,
                               template (present) SignatureType p_signature) := {

        sessionID := p_SessionID,
        notification := omit,
        signature_ := p_signature
    }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

N.2.1 V2G protocol handshake.

N.2.1.1 CMN templates for SupportedAppProtocol

```
module Templates_CMN_SupportedAppProtocol {

     import from urn_iso_15118_2_2010_AppProtocol language "XSD" all;
     import from XSDAUX all;

     template AppProtocolType md_CMN_CMN_AppProtocolType_001(
                              template (present) ProtocolNamespaceType p_protocolNamespace,
                              template (present) XSDAUX.unsignedInt p_versionNumberMajor,
                              template (present) XSDAUX.unsignedInt p_versionNumberMinor,
                              template (present) XSDAUX.unsignedByte p_schemaID,
                              template (present) PriorityType p_priority) := {

         protocolNamespace := p_protocolNamespace,
         versionNumberMajor := p_versionNumberMajor,
         versionNumberMinor := p_versionNumberMinor,
         schemaID := p_schemaID,
         priority := p_priority
     }

     template SupportedAppProtocolReq md_CMN_CMN_SupportedAppProtocolReq_001(
                                      template (present) AppProtocolType p_protocolList) := {

         appProtocol_list := {p_protocolList}
     }

     template SupportedAppProtocolRes md_CMN_CMN_SupportedAppProtocolRes_001(
                                      template (present) ResponseCodeType p_responseCode,
                                      template IdType p_sID) := {

         responseCode := p_responseCode, schemaID := p_sID
     }

    template SupportedAppProtocolRes md_CMN_CMN_SupportedAppProtocolRes_002(
                                     template (present) ResponseCodeType p_responseCode) := {

        responseCode := p_responseCode, schemaID := omit
    }
}
with {
 encode "ISO_V2G"
}
```

N.2.2 V2G messages

N.2.2.1 CMN templates for SessionSetup

```
module Templates_CMN_SessionSetup {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from XSDAUX language "XSD" {type long;}

    template BodyElement_group md_CMN_CMN_SessionSetupReq_001(
                               template (present) EvccIDType p_eVCCID) := {

        sessionSetupReq := {eVCCID := p_eVCCID}
    }

    template BodyElement_group md_CMN_CMN_SessionSetupRes_001(
                               template (present) ResponseCodeType p_resCode,
                               template (present) EvseIDType p_eVSEID,
                               template XSDAUX.long p_EVSETimeStamp) := {
        sessionSetupRes := {

                responseCode := p_resCode,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                eVSEID := p_eVSEID,
                eVSETimeStamp := p_EVSETimeStamp
        }
    }
}
```

N.2.2.2 CMN templates for ServiceDiscovery

```
module Templates_CMN_ServiceDiscovery {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;

    template BodyElement_group m_CMN_CMN_ServiceDiscoveryReq_001 := {
         serviceDiscoveryReq := {}
    }

    template BodyElement_group md_CMN_CMN_ServiceDiscoveryReq_002(
                              template ServiceCategoryType p_serviceCategory) := {

         serviceDiscoveryReq := {
             serviceCategory := p_serviceCategory
         }
    }

    template BodyElement_group md_CMN_CMN_ServiceDiscoveryReq_003(
                              template ServiceScopeType p_serviceScope,
                              template ServiceCategoryType p_serviceCategory) := {

        serviceDiscoveryReq := {
            serviceScope := p_serviceScope,
            serviceCategory := p_serviceCategory
        }
    }

    template BodyElement_group md_CMN_CMN_ServiceDiscoveryRes_001(
                                 template (present) PaymentOptionType p_paymentOption,
                              template (present) ServiceIDType p_serviceID,
                              template (present) boolean p_freeService,
                              template (present) EnergyTransferModeType p_transferMode,
                              template ServiceListType p_serviceList) := {

        serviceDiscoveryRes :=
        {
            responseCode := oK,
            paymentOptionList :=
            {
                paymentOption_list :=
                {
                    p_paymentOption
                }
            },
            chargeService :=
            {
                serviceID := p_serviceID,
                serviceName := omit,
                serviceCategory := eVCharging,
                serviceScope := omit,
                freeService := p_freeService,
                supportedEnergyTransferMode :=
                {
                    energyTransferMode_list :=
                    {
                        p_transferMode
                    }
                }
            },
            serviceList := p_serviceList
        }
     }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
template ServiceListType m_CMN_CMN_ServiceList_001() := {
    service_list := {
        {
            serviceType := {
                serviceID := 2,
                serviceName := "Certificate",
                serviceCategory := contractCertificate,
                serviceScope := omit,
                freeService := true
            }
        },
        {
            serviceType := {
                serviceID := 3,
                serviceName := "InternetAccess",
                serviceCategory := internet,
                serviceScope := omit,
                freeService := true
            }
        }
    }
}

template ServiceListType m_CMN_CMN_ServiceList_002() := {
    service_list := {
        {
            serviceType := {
                serviceID := 3,
                serviceName := "InternetAccess",
                serviceCategory := internet,
                serviceScope := omit,
                freeService := true
            }
        }
    }
}

template ServiceType m_CMN_CMN_ServiceType_001(in template (present) ServiceIDType p_serviceID,
                                               in template ServiceNameType p_serviceName,
                                               in    template    (present)    ServiceCategoryType
p_serviceCategory,
                                               in template ServiceScopeType p_serviceScope,
                                               in template (present) boolean p_freeService) := {

    serviceID := p_serviceID,
    serviceName := p_serviceName,
    serviceCategory := p_serviceCategory,
    serviceScope := p_serviceScope,
    freeService := p_freeService
}

template BodyElement_group md_CMN_CMN_ServiceDiscoveryRes_002(
                           template (present) ResponseCodeType p_responseCode,
                           template (present) PaymentOptionType p_paymentOption,
                           template (present) boolean p_freeService,
                           template (present) EnergyTransferModeType p_transferMode,
                           template(present) ServiceIDType p_serviceID) := {

    serviceDiscoveryRes :=
    {
        responseCode := p_responseCode,
        paymentOptionList :=
        {
            paymentOption_list :=
            {
                p_paymentOption
            }
        },
        chargeService :=
        {
            serviceID := p_serviceID,
            serviceName := omit,
            serviceCategory := eVCharging,
            serviceScope := omit,
            freeService := p_freeService,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            supportedEnergyTransferMode :=
            {
                energyTransferMode_list :=
                {
                    p_transferMode
                }
            }
        },
        serviceList := omit
    }
 }

template BodyElement_group md_CMN_CMN_ServiceDiscoveryRes_003(
                           template(present) ServiceIDType p_serviceID) := {

    serviceDiscoveryRes :=
    {
        responseCode := oK,
        paymentOptionList :=
        {
            paymentOption_list :=
            {
                contract,
                externalPayment
            }
        },
        chargeService :=
        {
            serviceID := p_serviceID,
            serviceName := omit,
            serviceCategory := eVCharging,
            serviceScope := omit,
            freeService := true,
            supportedEnergyTransferMode :=
            {
                energyTransferMode_list :=
                {
                    aC_single_phase_core, aC_three_phase_core
                }
            }
        },
        serviceList := omit
    }
 }

template BodyElement_group md_CMN_CMN_ServiceDiscoveryRes_004(
                           template (present) PaymentOptionType p_paymentOption,
                           template (present) ServiceIDType p_serviceID,
                           template (present) boolean p_freeService,
                           template (present) EnergyTransferModeType p_transferMode,
                           template ServiceListType p_serviceList) := {

    serviceDiscoveryRes :=
    {
        responseCode := oK,
        paymentOptionList :=
        {
            paymentOption_list :=
            {
                p_paymentOption
            }
        },
        chargeService :=
        {
            serviceID := p_serviceID,
            serviceName := omit,
            serviceCategory := eVCharging,
            serviceScope := omit,
            freeService := p_freeService,
            supportedEnergyTransferMode :=
            {
                energyTransferMode_list :=
                {
                    p_transferMode
                }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                }
            },
            serviceList := p_serviceList
        }
    }
}
```

N.2.2.3 CMN templates for Authorization

```
module Templates_CMN_Authorization {

    import from XSDAUX language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;

    //  V2G Body for AuthorizationReq EIM
    template BodyElement_group m_CMN_CMN_AuthorizationReq_001() := {

        authorizationReq := {
            genChallenge := omit,
            id := omit
        }
    }

    //  V2G Body for AuthorizationReq PnC
    template BodyElement_group md_CMN_CMN_AuthorizationReq_002(
                               template XSDAUX.ID p_id,
                               template GenChallengeType p_genChallenge) := {
        authorizationReq :=
        {
            id := p_id,
            genChallenge := p_genChallenge
        }
    }

    //  V2G Body for AuthorizationReq EIM/PnC
    template BodyElement_group mw_CMN_CMN_AuthorizationReq_003() := {

        authorizationReq :=
        {
            id := *,
            genChallenge := *
        }
    }

    //  V2G Body for AuthorizationRes
    template BodyElement_group md_CMN_CMN_AuthorizationRes_001(
                               template(present) ResponseCodeType p_responseCode,
                               template(present) EVSEProcessingType p_evseproc) := {

        authorizationRes :=
        {
            eVSEProcessing := p_evseproc,
            responseCode := p_responseCode
        }
    }
}
```

N.2.2.4 CMN templates for PowerDelivery

```
module Templates_CMN_PowerDelivery {

    import from XSDAUX language "XSD" { type Boolean; }
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;

    type enumerated ePowerDeliveryMode {e_Mode_PowerDelivery_Start(1),
                                        e_Mode_PowerDelivery_Stop(2)}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
type enumerated V2G_TC_PowerDelivery_Modes {e_V2G_HeaderType, e_V2G_SessionID,
                                            e_V2G_Wrong_SessionID, e_V2G_Message }

template BodyElement_group md_CMN_AC_PowerDeliveryReq_001(
                           template(present) ChargeProgressType p_chargeProgress,
                           template(present) SAIDType p_sAScheduleTupleID) := {
    powerDeliveryReq := {

        chargeProgress := p_chargeProgress,
        sAScheduleTupleID := p_sAScheduleTupleID,
        chargingProfile := omit,
        eVPowerDeliveryParameter := omit
    }
}

template BodyElement_group md_CMN_DC_PowerDeliveryReq_002(
                           template(present) ChargeProgressType p_chargeProgress,
                           template(present) SAIDType p_sAScheduleTupleID,
                           template ChargingProfileType p_chargingProfile,
                           template(present) DC_EVPowerDeliveryParameter
                           p_dcEVPowerDeliveryParameter) := {
     powerDeliveryReq :=
     {
         chargeProgress := p_chargeProgress,
         sAScheduleTupleID := p_sAScheduleTupleID,
         chargingProfile := p_chargingProfile,
         eVPowerDeliveryParameter := {
             dC_EVPowerDeliveryParameter := p_dcEVPowerDeliveryParameter
         }
     }
 }

 template BodyElement_group mdw_CMN_DC_PowerDeliveryReq_003(
                           template(present) ChargeProgressType p_chargeProgress,
                           template(present) SAIDType p_sAScheduleTupleID,
                           template ChargingProfileType p_chargingProfile) := {
     powerDeliveryReq :=
     {
         chargeProgress := p_chargeProgress,
         sAScheduleTupleID := p_sAScheduleTupleID,
         chargingProfile := p_chargingProfile,
         eVPowerDeliveryParameter := *
     }
 }

 template BodyElement_group md_CMN_AC_PowerDeliveryReq_003(
                            template(present) ChargeProgressType p_enablePower,
                            template(present) SAIDType p_sAScheduleTupleID,
                            template ChargingProfileType p_CProfile) := {

    powerDeliveryReq :=
    {
      chargeProgress := p_enablePower,
      sAScheduleTupleID := p_sAScheduleTupleID,
      chargingProfile := p_CProfile,
      eVPowerDeliveryParameter := omit
    }
 }

 template BodyElement_group md_CMN_CMN_PowerDeliveryRes_001(
                            template(present) ResponseCodeType p_responseCode,
                            template(present) EVSEStatus_group p_evsestatus) := {

      powerDeliveryRes :=
      {
          responseCode := p_responseCode,
          eVSEStatus := p_evsestatus
      }
 }

 template AC_EVSEStatusType md_CMN_AC_EVSEStatus_001(
                            template (present) Boolean v_rCD,
                            template (present) XSDAUX.unsignedShort v_notificationMaxDelay,
                            template (present) EVSENotificationType v_eVSENotification) := {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        rCD := v_rCD,
        notificationMaxDelay := v_notificationMaxDelay,
        eVSENotification := v_eVSENotification
    }

    template DC_EVSEStatusType md_CMN_DC_EVSEStatus_001(
                               template(present) XSDAUX.unsignedShort v_notificationMaxDelay,
                              template(present) EVSENotificationType v_eVSENotification,
                              template IsolationLevelType v_eVSEIsolationStatus,
                              template(present) DC_EVSEStatusCodeType v_eVSEStatusCode) := {

        notificationMaxDelay := v_notificationMaxDelay,
        eVSENotification := v_eVSENotification,
        eVSEIsolationStatus := v_eVSEIsolationStatus,
        eVSEStatusCode := v_eVSEStatusCode
    }

    template EVSEStatus_group md_CMN_AC_EVSEStatus_002(
                              template(present) AC_EVSEStatusType status) := {

       aC_EVSEStatus := {
        rCD := status.rCD,
        notificationMaxDelay := status.notificationMaxDelay,
        eVSENotification := status.eVSENotification
       }
    }

    template EVSEStatus_group md_CMN_DC_EVSEStatus_002(
                              template(present) DC_EVSEStatusType status) := {

      dC_EVSEStatus := {
          notificationMaxDelay :=status.notificationMaxDelay,
          eVSENotification :=status.eVSENotification,
          eVSEIsolationStatus :=status.eVSEIsolationStatus,
          eVSEStatusCode  :=status.eVSEStatusCode
      }
    }

    template DC_EVPowerDeliveryParameterType md_CMN_DC_EVPowerDeliveryParameter(
                                              template(present) DC_EVStatusType dC_EVStatus,
                                             template Boolean bulkChargingComplete,
                                             template(present) Boolean chargingComplete) := {
      dC_EVStatus          := dC_EVStatus,
      bulkChargingComplete := bulkChargingComplete,
      chargingComplete     := chargingComplete
    }

    private type record length (1..1024) of ProfileEntryType ProfileEntryTypeList;

    template ProfileEntryType md_CMN_CMN_ProfileEntry_001(
                              template(present) XSDAUX.unsignedInt entryStart,
                             template(present) PhysicalValueType maxPower,
                             template MaxNumPhasesType maxNumPhases) := {

       chargingProfileEntryStart := entryStart,
       chargingProfileEntryMaxPower := maxPower,
       chargingProfileEntryMaxNumberOfPhasesInUse := maxNumPhases
    }

    template ChargingProfileType md_CMN_CMN_ChargingProfile_001(
                                 template(present) ProfileEntryTypeList profileList) := {
       profileEntry_list := profileList
    }
}
```

N.2.2.5 CMN templates for SessionStop

```
module Templates_CMN_SessionStop {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from XSDAUX language "XSD" {type long;}

    template BodyElement_group md_CMN_CMN_SessionStopReq_001(
                               template(present) ChargingSessionType p_chargingSession) := {
        sessionStopReq :=
        {
            chargingSession := p_chargingSession
        }
    }

    template BodyElement_group md_CMN_CMN_SessionStopRes_001(
                               template(present) ResponseCodeType p_resCode) := {
        sessionStopRes :=
        {
            responseCode := p_resCode
        }
    }
}
```

N.2.2.6 CMN templates for ChargingStatus

```
module Templates_CMN_ChargingStatus {

   import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
   import from XSDAUX language "XSD" all;
   import from urn_iso_15118_2_2013_MsgBody language "XSD" all;

   template BodyElement_group m_CMN_AC_ChargingStatusReq_001 := {
      chargingStatusReq := {}
   }

   template BodyElement_group md_CMN_AC_ChargingStatusRes_001(
                              template (present) ResponseCodeType p_responseCode,
                              template (present) EvseIDType p_evseID,
                              template (present) SAIDType p_sAScheduleTuple,
                              template PhysicalValueType p_eVSEMaxCurrent,
                              template MeterInfoType p_meterInfo,
                              template Boolean p_receiptRequired,
                              template (present) AC_EVSEStatusType p_aCeVSEStatus) := {

      chargingStatusRes := {

        responseCode := p_responseCode,
        eVSEID := p_evseID,
        sAScheduleTupleID := p_sAScheduleTuple,
        eVSEMaxCurrent := p_eVSEMaxCurrent,
        meterInfo := p_meterInfo,
        receiptRequired := p_receiptRequired,
        aC_EVSEStatus := p_aCeVSEStatus
       }
   }

   template MeterInfoType md_CMN_AC_meterInfoType(
                          template (present) MeterIDType p_meterID,
                          template XSDAUX.unsignedLong p_meterReading,
                          template SigMeterReadingType p_sigMeterReading,
                          template MeterStatusType p_meterStatus,
                          template XSDAUX.long p_tMeter) := {

            meterID := p_meterID,
            meterReading := p_meterReading,
            sigMeterReading := p_sigMeterReading,
            meterStatus := p_meterStatus,
            tMeter := p_tMeter
     }
 }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

N.2.2.7 CMN templates for CableCheck

```
module Templates_CMN_CableCheck {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from XSDAUX language "XSD" all;

    template DC_EVStatusType md_CMN_DC_EVStatus_001(
                            template(present) XSDAUX.Boolean p_eVReady,
                            template(present) DC_EVErrorCodeType p_eVErrorCode,
                            template(present) PercentValueType p_eVRESSSOC):= {

                eVReady := p_eVReady,
                eVErrorCode := p_eVErrorCode,
                eVRESSSOC := p_eVRESSSOC
    }

    template DC_EVSEStatusType md_CMN_DC_EVSEStatus_001(
                              template(present) XSDAUX.unsignedShort v_notificationMaxDelay,
                              template(present) EVSENotificationType v_eVSENotification,
                              template IsolationLevelType v_eVSEIsolationStatus,
                              template(present) DC_EVSEStatusCodeType v_eVSEStatusCode) := {

            notificationMaxDelay := v_notificationMaxDelay,
            eVSENotification := v_eVSENotification,
            eVSEIsolationStatus := v_eVSEIsolationStatus,
            eVSEStatusCode := v_eVSEStatusCode
    }

    template BodyElement_group md_CMN_DC_CableCheckReq_001(
                              template (present) DC_EVStatusType p_DC_EVStatus):= {
            cableCheckReq :=
            {
                dC_EVStatus := p_DC_EVStatus

            }
    }

    template BodyElement_group md_CMN_DC_CableCheckRes_001(
                              template(present) ResponseCodeType p_responseCode,
                              template(present) EVSEProcessingType p_EVSEProcessing,
                              template(present) DC_EVSEStatusType p_DC_EVSEStatus):= {

            cableCheckRes:=
            {
                responseCode := p_responseCode,
                eVSEProcessing := p_EVSEProcessing,
                dC_EVSEStatus := p_DC_EVSEStatus
            }
    }
}
```

N.2.2.8 CMN templates for PreCharge

```
module Templates_CMN_PreCharge {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from XSDAUX language "XSD" all;

    template BodyElement_group md_CMN_DC_PreChargeReq_001(
                              template(present) DC_EVStatusType p_dC_EVStatus,
                              template(present) PhysicalValueType p_eVTargetVoltage,
                              template(present) PhysicalValueType p_eVTargetCurrent) := {

        preChargeReq :=
        {
            dC_EVStatus := p_dC_EVStatus,
            eVTargetVoltage := p_eVTargetVoltage,
            eVTargetCurrent := p_eVTargetCurrent
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        }
    }

    template BodyElement_group md_CMN_DC_PreChargeRes_001(
                              template (present) ResponseCodeType p_resCode,
                               template (present) DC_EVSEStatusType p_dC_EVSEStatus,
                               template (present) PhysicalValueType p_eVSEPresentVoltage) := {

        preChargeRes :=
        {
             responseCode := p_resCode,
            dC_EVSEStatus := p_dC_EVSEStatus,
            eVSEPresentVoltage := p_eVSEPresentVoltage

        }
    }
}
```

N.2.2.9 CMN templates for CurrentDemand

```
module Templates_CMN_CurrentDemand {

   import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
   import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
   import from XSDAUX language "XSD" all;

   template PhysicalValueType md_CMN_DC_remainingTimeToFullSoC(
                              UnitMultiplierType p_multiplier,
                              UnitSymbolType p_unit,
                              XSDAUX.short p_value_):= {

                  multiplier := p_multiplier,
                  unit := p_unit,
                  value_ := p_value_
   }


   template PhysicalValueType md_CMN_DC_remainingTimeToBulkSoC(
                              UnitMultiplierType p_multiplier,
                              UnitSymbolType p_unit,
                              XSDAUX.short p_value_):= {

                  multiplier := p_multiplier,
                  unit := p_unit,
                  value_ := p_value_
   }

   template MeterInfoType md_CMN_DC_meterInfoType(
                        template(present) MeterIDType v_meterID,
                        template(present) XSDAUX.unsignedLong v_meterReading,
                        template(present) SigMeterReadingType v_sigMeterReading,
                        template(present) MeterStatusType v_meterStatus,
                        template(present) XSDAUX.long v_tMeter):= {

              meterID := v_meterID,
              meterReading := v_meterReading,
              sigMeterReading    := v_sigMeterReading,
              meterStatus    := v_meterStatus,
              tMeter := v_tMeter
   }

   template BodyElement_group md_CMN_DC_CurrentDemandReq_001(
                              template(present) DC_EVStatusType p_dC_EVStatus,
                              template(present) PhysicalValueType p_eVTargetCurrent,
                              template PhysicalValueType p_eVMaximumVoltageLimit,
                              template PhysicalValueType p_eVMaximumCurrentLimit,
                              template PhysicalValueType p_eVMaximumPowerLimit,
                              template XSDAUX.Boolean p_bulkChargingComplete,
                              template(present) XSDAUX.Boolean p_chargingComplete,
                              template PhysicalValueType p_remainingTimeToFullSoC,
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                template PhysicalValueType p_remainingTimeToBulkSoC,
                                template(present) PhysicalValueType p_eVTargetVoltage) := {

            currentDemandReq := {

                dC_EVStatus := p_dC_EVStatus,
                eVTargetCurrent := p_eVTargetCurrent,
                eVMaximumVoltageLimit := p_eVMaximumVoltageLimit,
                eVMaximumCurrentLimit := p_eVMaximumCurrentLimit,
                eVMaximumPowerLimit := p_eVMaximumPowerLimit,
                bulkChargingComplete := p_bulkChargingComplete,
                chargingComplete := p_chargingComplete,
                remainingTimeToFullSoC := p_remainingTimeToFullSoC,
                remainingTimeToBulkSoC := p_remainingTimeToBulkSoC,
                eVTargetVoltage := p_eVTargetVoltage
            }
    }

    template BodyElement_group md_CMN_DC_CurrentDemandRes_001(
                               template (present) ResponseCodeType p_responseCode,
                               template (present) DC_EVSEStatusType p_dC_EVSEStatus,
                               template (present) PhysicalValueType p_eVSEPresentVoltage,
                               template (present) PhysicalValueType p_eVSEPresentCurrent,
                               template (present) XSDAUX.Boolean p_eVSECurrentLimitAchieved,
                               template (present) XSDAUX.Boolean p_eVSEVoltageLimitAchieved,
                               template (present) XSDAUX.Boolean p_eVSEPowerLimitAchieved,
                               template PhysicalValueType p_eVSEMaximumVoltageLimit,
                               template PhysicalValueType p_eVSEMaximumCurrentLimit,
                               template PhysicalValueType p_eVSEMaximumPowerLimit,
                               template (present) EvseIDType    p_eVSEID,
                               template (present) SAIDType p_sAScheduleTupleID,
                               template MeterInfoType p_meterInfo,
                               template XSDAUX.Boolean p_receiptRequired):= {

                currentDemandRes := {

                           responseCode := p_responseCode,
                        dC_EVSEStatus := p_dC_EVSEStatus,
                           eVSEPresentVoltage := p_eVSEPresentVoltage,
                        eVSEPresentCurrent := p_eVSEPresentCurrent,
                        eVSECurrentLimitAchieved := p_eVSECurrentLimitAchieved,
                        eVSEVoltageLimitAchieved := p_eVSEVoltageLimitAchieved,
                        eVSEPowerLimitAchieved := p_eVSEPowerLimitAchieved,
                        eVSEMaximumVoltageLimit := p_eVSEMaximumVoltageLimit,
                        eVSEMaximumCurrentLimit    := p_eVSEMaximumCurrentLimit,
                        eVSEMaximumPowerLimit := p_eVSEMaximumPowerLimit,
                        eVSEID := p_eVSEID,
                        sAScheduleTupleID := p_sAScheduleTupleID,
                        meterInfo := p_meterInfo,
                        receiptRequired    := p_receiptRequired
                }
     }
}
```

N.2.2.10 CMN templates for WeldingDetection

```
module Templates_CMN_WeldingDetection {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from XSDAUX language "XSD" all;

    template BodyElement_group md_CMN_DC_WeldingDetectionReq_001(
                                     template (present) DC_EVStatusType p_dCeVStatus):= {

        weldingDetectionReq :=
        {
            dC_EVStatus := p_dCeVStatus

        }
    }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    template BodyElement_group md_CMN_DC_WeldingDetectionRes_001(
                               template(present) ResponseCodeType p_responseCode,
                               template(present) DC_EVSEStatusType    p_dC_EVSEStatus,
                               template(present) PhysicalValueType    p_eVSEPresentVoltage):= {

        weldingDetectionRes :=
        {
            responseCode         := p_responseCode,
            dC_EVSEStatus         := p_dC_EVSEStatus,
            eVSEPresentVoltage     := p_eVSEPresentVoltage
        }
    }
}
```

N.3 SECC templates

N.3.1 V2G protocol handshake

N.3.1.1 SECC templates for SupportedAppProtocol

```
module Templates_SECC_SupportedAppProtocol {

    public import from urn_iso_15118_2_2010_AppProtocol language "XSD" all;

    template ResponseCodeType m_SECC_CMN_validResponseCodes() := (
                              oK_SuccessfulNegotiation, oK_SuccessfulNegotiationWithMinorDeviation);

}
```

N.3.2 V2G messages

N.3.2.1 SECC templates for ServiceDiscovery

```
module Templates_SECC_ServiceDiscovery {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;

    template BodyElement_group mdw_SECC_CMN_ServiceDiscoveryRes_001(
                               template (present) ResponseCodeType p_responseCode) := {

            serviceDiscoveryRes :=
            {
                responseCode := p_responseCode,
                paymentOptionList := ?,
                chargeService := ?,
                serviceList := omit
            }
    }

    template BodyElement_group md_SECC_CMN_ServiceDiscoveryRes_002(
                               template (present) ResponseCodeType p_responseCode,
                               template (present) PaymentOptionListType p_paymentOptions,
                               template (present) ChargeServiceType p_chargeService,
                               template  ServiceListType p_serviceList) := {

        serviceDiscoveryRes := {
            responseCode := p_responseCode,
            paymentOptionList := p_paymentOptions,
            chargeService := p_chargeService,
            serviceList := p_serviceList
        }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }
}
```

N.3.2.2 SECC templates for ServiceDetail

```
module Templates_SECC_ServiceDetail {

       import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
       import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
       import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
       import from XSDAUX language "XSD" all;

       template BodyElement_group md_SECC_CMN_ServiceDetailReq_001(
                                   template(present) ServiceIDType p_serviceID) := {

         serviceDetailReq := {
             serviceID := p_serviceID
             }
       }

       template BodyElement_group mdw_SECC_CMN_ServiceDetailRes_001(
                                   template(present) ServiceIDType p_serviceID) := {

         serviceDetailRes := {
             responseCode := oK,
             serviceID := p_serviceID,
             serviceParameterList := ?
         }
      }

      template BodyElement_group mw_SECC_CMN_ServiceDetailRes_002() := {
        serviceDetailRes := {
            responseCode := ?,
            serviceID := ?,
            serviceParameterList := ?
        }
     }

     template BodyElement_group mdw_SECC_CMN_ServiceDetailRes_003(
                                template (present) ResponseCodeType p_responseCode) := {

       serviceDetailRes := {
           responseCode := p_responseCode,
           serviceID := ?,
           serviceParameterList := omit
      }
   }
}
```

N.3.2.3 SECC templates for PaymentServiceSelection

```
module Templates_SECC_PaymentServiceSelection {

   import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
   import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
   import from XSDAUX language "XSD" all;

   template BodyElement_group md_SECC_CMN_PaymentServiceSelectionReq_001(
                                template (present) PaymentOptionType p_selPaymentOption,
                              template (present) ServiceIDType p_serviceID) := {

      paymentServiceSelectionReq :=
       {
        selectedPaymentOption := p_selPaymentOption,
        selectedServiceList :=
         {
          selectedService_list :=
           {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
          {
           serviceID := p_serviceID,
           parameterSetID := omit
          }
        }
      }
    }
}

template BodyElement_group md_SECC_CMN_PaymentServiceSelectionReq_002(
                            template (present) PaymentOptionType p_selPaymentOption,
                           template (present) ServiceIDType p_serviceID,
                           template XSDAUX.short p_parameterSetID) := {

   paymentServiceSelectionReq :=
    {
     selectedPaymentOption := p_selPaymentOption,
     selectedServiceList :=
      {
       selectedService_list :=
        {
          {
           serviceID := p_serviceID,
           parameterSetID := p_parameterSetID
          }
        }
      }
    }
 }

 template BodyElement_group md_SECC_CMN_PaymentServiceSelectionReq_003(
                            template (present) PaymentOptionType p_selPaymentOption,
                            template (present) ServiceIDType p_serviceID,
                            template XSDAUX.short parameterSetID) := {

    paymentServiceSelectionReq :=
     {
      selectedPaymentOption := p_selPaymentOption,
      selectedServiceList :=
       {
        selectedService_list :=
         {
           {
            serviceID := 1,
            parameterSetID := omit
           },
           {
            serviceID := p_serviceID,
            parameterSetID := parameterSetID
           }
         }
       }
    }
 }

 template BodyElement_group md_SECC_CMN_PaymentServiceSelectionReq_004(
                            template (present) PaymentOptionType p_selPaymentOption,
                            template (present) ServiceIDType p_serviceID,
                            template XSDAUX.short parameterSetID,
                            template (present) ServiceIDType p_serviceIDCustomVAS,
                            template XSDAUX.short parameterSetIDCustomVAS) := {

    paymentServiceSelectionReq :=
     {
      selectedPaymentOption := p_selPaymentOption,
      selectedServiceList :=
       {
        selectedService_list :=
         {
           {
            serviceID := 1,
            parameterSetID := omit
           },
           {
            serviceID := p_serviceID,
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                parameterSetID := parameterSetID
               },
               {
                serviceID := p_serviceIDCustomVAS,
                parameterSetID := parameterSetIDCustomVAS
               }
             }
           }
        }
    }

    template BodyElement_group md_SECC_CMN_PaymentServiceSelectionRes_001(
                              template (present) ResponseCodeType p_responseCode) := {

      paymentServiceSelectionRes :=
        {
         responseCode := p_responseCode
        }
    }
 }
```

N.3.2.4 SECC templates for PaymentDetails

```
module Templates_SECC_PaymentDetails {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from XSDAUX language "XSD" all;

    template BodyElement_group md_SECC_CMN_PaymentDetailsReq_001(
                              template(present) EMAIDType_1 p_eMAID,
                              template(present) CertificateChainType p_certificateChainType) := {

        paymentDetailsReq := {
            eMAID := p_eMAID,
            contractSignatureCertChain := p_certificateChainType
        }
    }

    template BodyElement_group md_SECC_CMN_PaymentDetailsRes_001(
                              template (present) ResponseCodeType p_responseCode,
                              template(present) GenChallengeType _genChallenge,
                              template(present) XSDAUX.long _eVSETimeStamp) := {

        paymentDetailsRes := {
            responseCode := p_responseCode,
            genChallenge := _genChallenge,
            eVSETimeStamp := _eVSETimeStamp
        }
    }

    template BodyElement_group mdw_SECC_CMN_PaymentDetailsRes_002(
                              template (present) ResponseCodeType p_responseCode) := {

        paymentDetailsRes := {
            responseCode := p_responseCode,
            genChallenge := ?,
            eVSETimeStamp := ?
        }
    }

    template ResponseCodeType m_CMN_OkResponseCodeList_001() :=
                            (oK, oK_CertificateExpiresSoon);
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

N.3.2.5 SECC templates for ChargeParameterDiscovery

```
module Templates_SECC_ChargeParameterDiscovery {

    import from DataStructure_V2G all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from XSDAUX language "XSD" all;

    template BodyElement_group md_SECC_AC_ChargeParameterDiscoveryReq_001(
                              template XSDAUX.unsignedShort
                              p_maxEntriesSAScheduleTuple,
                              template(present) EnergyTransferModeType
                              p_requestedEnergyTransferMode,
                              template XSDAUX.unsignedInt p_departureTime,
                              template(present) PhysicalValueType p_eAmount,
                              template(present) PhysicalValueType p_eVMaxVoltage,
                              template(present) PhysicalValueType p_eVMaxCurrent,
                              template(present) PhysicalValueType p_eVMinCurrent) := {

        chargeParameterDiscoveryReq := {
            maxEntriesSAScheduleTuple := p_maxEntriesSAScheduleTuple,
            requestedEnergyTransferMode := p_requestedEnergyTransferMode,
            eVChargeParameter := {
                aC_EVChargeParameter := {
                    departureTime := p_departureTime,
                    eAmount := p_eAmount,
                    eVMaxVoltage := p_eVMaxVoltage,
                    eVMaxCurrent := p_eVMaxCurrent,
                    eVMinCurrent := p_eVMinCurrent
                }
            }
        }
    }

    template BodyElement_group md_SECC_AC_ChargeParameterDiscoveryReq_002(
                              template(present) EnergyTransferModeType
                              p_requestedEnergyTransferMode) := {

        chargeParameterDiscoveryReq := {
            maxEntriesSAScheduleTuple := omit,
            requestedEnergyTransferMode := p_requestedEnergyTransferMode,
            eVChargeParameter := {
                aC_EVChargeParameter := {
                    departureTime := omit,
                    eAmount := {multiplier := 0, unit := w, value_ := 100},
                    eVMaxVoltage := {
                        multiplier := 0, unit := v, value_ := 400
                    },
                    eVMaxCurrent := {
                        multiplier := 0, unit := a, value_ := -100
                    },
                    eVMinCurrent := {multiplier := 0, unit := a, value_ := 500}
                }
            }
        }
    }

    template BodyElement_group mdw_SECC_CMN_ChargeParameterDiscoveryRes_001(
                              template (present) ResponseCodeType p_responseCode) := {

        chargeParameterDiscoveryRes := {
            responseCode := p_responseCode,
            eVSEProcessing := ?,
            sASchedules := omit,
            eVSEChargeParameter := ?
        }
    }

    template BodyElement_group mdw_SECC_AC_ChargeParameterDiscoveryRes_001(
                              template(present) EVSEProcessingType p_evseproc,
                              template(present) AC_EVSEStatusType  p_aCeVSEStatus,
                              template SalesTariffType p_salesTariff) := {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    chargeParameterDiscoveryRes := {
        responseCode := oK,
        eVSEProcessing := p_evseproc,
        sASchedules := {
            sAScheduleList := {
                {
                    mdw_saScheduleTuple_001(p_salesTariff),
                    *,    //optional second SalesTariff
                    *    //optional third SalesTariff
                }
            }
          },
        eVSEChargeParameter := {
            aC_EVSEChargeParameter := {
                aC_EVSEStatus := p_aCeVSEStatus,
                eVSENominalVoltage := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMaxCurrent := {
                    multiplier := ?, unit := ?, value_ := ?
                }
            }
        }
    }
}

template urn_iso_15118_2_2013_MsgDataTypes.SAScheduleTupleType mdw_saScheduleTuple_001(
                                        template SalesTariffType p_salesTariff) := {
      sAScheduleTupleID := ?,
      pMaxSchedule := ?,
      salesTariff := p_salesTariff
}

template SAScheduleListType md_SECC_CMN_SASchedules_001(
                            in SAIDType p_sAScheduleTupleID,
                            in DurationType v_duration,
                            in PhysicalValueType p_pMax) := {

    sAScheduleTuple_list := {
        {
            sAScheduleTupleID := p_sAScheduleTupleID,
            pMaxSchedule := {
                pMaxScheduleEntry_list := {
                    {
                        timeInterval := {
                            relativeTimeInterval := {
                                start_ := 0,
                                duration := v_duration
                            }
                        },
                        pMax := p_pMax
                    }
                }
            },
            salesTariff := omit
        }
    }
}

template BodyElement_group mdw_SECC_AC_ChargeParameterDiscoveryRes_002(
                            template(present) AC_EVSEStatusType  p_aCeVSEStatus) := {

    chargeParameterDiscoveryRes := {
        responseCode := oK,
        eVSEProcessing := ongoing,
        sASchedules := omit,
        eVSEChargeParameter := {
        aC_EVSEChargeParameter := {
            aC_EVSEStatus := p_aCeVSEStatus,
            eVSENominalVoltage := {
                multiplier := ?, unit := ?, value_ := ?
            },
            eVSEMaxCurrent := {
                multiplier := ?, unit := ?, value_ := ?
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            }
        }
    }
  }
}

template BodyElement_group mw_SECC_AC_ChargeParameterDiscoveryRes_003() := {

    chargeParameterDiscoveryRes := {
        responseCode := oK,
        eVSEProcessing := ongoing,
        sASchedules := *,
        eVSEChargeParameter := {
        aC_EVSEChargeParameter := {
            aC_EVSEStatus := ?,
            eVSENominalVoltage := {
                multiplier := ?, unit := ?, value_ := ?
            },
            eVSEMaxCurrent := {
                multiplier := ?, unit := ?, value_ := ?
            }
        }
    }
  }
}

template BodyElement_group mw_SECC_AC_ChargeParameterDiscoveryRes_004() := {

    chargeParameterDiscoveryRes := {
        responseCode := oK,
        eVSEProcessing := finished,
        sASchedules := *,
        eVSEChargeParameter := {
            aC_EVSEChargeParameter := {
                aC_EVSEStatus := ?,
                eVSENominalVoltage := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMaxCurrent := {
                    multiplier := ?, unit := ?, value_ := ?
                }
            }
        }
    }
}

template BodyElement_group md_SECC_DC_ChargeParameterDiscoveryReq_001(
                           template XSDAUX.unsignedShort
                           p_maxEntriesSAScheduleTuple,
                           template(present) EnergyTransferModeType
                           p_requestedEnergyTransferMode,
                           template XSDAUX.unsignedInt p_departureTime,
                           template(present) DC_EVStatusType p_dC_EVStatus,
                           template(present) PhysicalValueType p_eVMaximumCurrentLimit,
                           template PhysicalValueType p_eVMaximumPowerLimit,
                           template(present) PhysicalValueType p_eVMaximumVoltageLimit) := {

    chargeParameterDiscoveryReq := {
        maxEntriesSAScheduleTuple := p_maxEntriesSAScheduleTuple,
        requestedEnergyTransferMode := p_requestedEnergyTransferMode,
        eVChargeParameter := {
            dC_EVChargeParameter := {
                departureTime := p_departureTime,
                dC_EVStatus := p_dC_EVStatus,
                eVMaximumCurrentLimit := p_eVMaximumCurrentLimit,
                eVMaximumPowerLimit := p_eVMaximumPowerLimit,
                eVMaximumVoltageLimit := p_eVMaximumVoltageLimit,
                eVEnergyCapacity := omit,
                eVEnergyRequest := omit,
                fullSOC := omit,
                bulkSOC := omit
            }
        }
    }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
template BodyElement_group md_SECC_DC_ChargeParameterDiscoveryReq_002(
                           template(present) EnergyTransferModeType
                           p_requestedEnergyTransferMode) := {

    chargeParameterDiscoveryReq := {
        maxEntriesSAScheduleTuple := omit,
        requestedEnergyTransferMode := p_requestedEnergyTransferMode,
        eVChargeParameter := {
            dC_EVChargeParameter := {
                departureTime := omit,
                dC_EVStatus := {
                    eVReady := false,
                    eVErrorCode := nO_ERROR,
                    eVRESSSOC := 50
                },
                eVMaximumCurrentLimit := {
                    multiplier := 0, unit := a, value_ := -100
                },
                eVMaximumPowerLimit := omit,
                eVMaximumVoltageLimit := {
                    multiplier := 0, unit := v, value_ := 400
                },
                eVEnergyCapacity := omit,
                eVEnergyRequest := omit,
                fullSOC := omit,
                bulkSOC := omit
            }
        }
    }
}

template BodyElement_group mdw_SECC_DC_ChargeParameterDiscoveryRes_001(
                           template(present) EVSEProcessingType p_evseproc,
                           template(present) EVSENotificationType p_eVSENotification,
                           template(present) DC_EVSEStatusCodeType p_eVSEStatusCode,
                           template SalesTariffType  p_salesTariff) := {

    chargeParameterDiscoveryRes := {
        responseCode := oK,
        eVSEProcessing := p_evseproc,
        sASchedules := {
            sAScheduleList := {
                {
                    mdw_saScheduleTuple_001(p_salesTariff),
                    *,    //optional second SalesTariff
                    *    //optional third SalesTariff
                }
            }
        },
        eVSEChargeParameter := {
            dC_EVSEChargeParameter := {
                dC_EVSEStatus := {
                    notificationMaxDelay := ?,
                    eVSENotification := p_eVSENotification,
                    eVSEIsolationStatus := *,
                    eVSEStatusCode := p_eVSEStatusCode
                },
                eVSEMaximumCurrentLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMaximumPowerLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMaximumVoltageLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMinimumCurrentLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMinimumVoltageLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSECurrentRegulationTolerance := *,
                eVSEPeakCurrentRipple := {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEEnergyToBeDelivered := *
            }
        }
    }
}

template BodyElement_group mdw_SECC_DC_ChargeParameterDiscoveryRes_002(
                           template(present) EVSENotificationType p_eVSENotification,
                           template(present) DC_EVSEStatusCodeType p_eVSEStatusCode) := {

    chargeParameterDiscoveryRes := {
        responseCode := oK,
        eVSEProcessing := ongoing,
        sASchedules := omit,
        eVSEChargeParameter := {
            dC_EVSEChargeParameter := {
                dC_EVSEStatus := {
                    notificationMaxDelay := ?,
                    eVSENotification := p_eVSENotification,
                    eVSEIsolationStatus := *,
                    eVSEStatusCode := p_eVSEStatusCode
                },
                eVSEMaximumCurrentLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMaximumPowerLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMaximumVoltageLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMinimumCurrentLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMinimumVoltageLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSECurrentRegulationTolerance := *,
                eVSEPeakCurrentRipple := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEEnergyToBeDelivered := *
            }
        }
    }
}

template BodyElement_group mw_SECC_DC_ChargeParameterDiscoveryRes_003() := {

    chargeParameterDiscoveryRes := {
        responseCode := oK,
        eVSEProcessing := ongoing,
        sASchedules := *,
        eVSEChargeParameter := {
            dC_EVSEChargeParameter := {
                dC_EVSEStatus := {
                    notificationMaxDelay := ?,
                    eVSENotification := ?,
                    eVSEIsolationStatus := *,
                    eVSEStatusCode := ?
                },
                eVSEMaximumCurrentLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMaximumPowerLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMaximumVoltageLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
                eVSEMinimumCurrentLimit := {
                    multiplier := ?, unit := ?, value_ := ?
                },
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                    eVSEMinimumVoltageLimit := {
                        multiplier := ?, unit := ?, value_ := ?
                    },
                    eVSECurrentRegulationTolerance := *,
                    eVSEPeakCurrentRipple := {
                        multiplier := ?, unit := ?, value_ := ?
                    },
                    eVSEEnergyToBeDelivered := *
                }
            }
        }
    }

    template BodyElement_group mw_SECC_DC_ChargeParameterDiscoveryRes_004() := {

        chargeParameterDiscoveryRes := {
            responseCode := oK,
            eVSEProcessing := finished,
            sASchedules := *,
            eVSEChargeParameter := {
                dC_EVSEChargeParameter := {
                    dC_EVSEStatus := {
                        notificationMaxDelay := ?,
                        eVSENotification := ?,
                        eVSEIsolationStatus := *,
                        eVSEStatusCode := ?
                    },
                    eVSEMaximumCurrentLimit := {
                        multiplier := ?, unit := ?, value_ := ?
                    },
                    eVSEMaximumPowerLimit := {
                        multiplier := ?, unit := ?, value_ := ?
                    },
                    eVSEMaximumVoltageLimit := {
                        multiplier := ?, unit := ?, value_ := ?
                    },
                    eVSEMinimumCurrentLimit := {
                        multiplier := ?, unit := ?, value_ := ?
                    },
                    eVSEMinimumVoltageLimit := {
                        multiplier := ?, unit := ?, value_ := ?
                    },
                    eVSECurrentRegulationTolerance := *,
                    eVSEPeakCurrentRipple := {
                        multiplier := ?, unit := ?, value_ := ?
                    },
                    eVSEEnergyToBeDelivered := *
                }
            }
        }
    }
}
```

N.3.2.6 SECC templates for CertificateUpdate

```
module Templates_SECC_CertificateUpdate {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from XSDAUX language "XSD" all;

    template BodyElement_group md_SECC_CMN_CertificateUpdateReq_001(
                              in template(present) XSDAUX.ID p_id,
                              in template(present) CertificateChainType p_contractSignatureCertChain,
                              in template(present) EMAIDType_1 p_eMAID,
                              in template(present) XSDAUX.string p_x509IssuerName,
                              in template(present) XSDAUX.integerXSD p_x509SerialNumber) := {

        certificateUpdateReq := {
            id := p_id,
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            contractSignatureCertChain := p_contractSignatureCertChain,
            eMAID := p_eMAID,
            listOfRootCertificateIDs := {
                rootCertificateID_list := {
                    {x509IssuerName := p_x509IssuerName,
                     x509SerialNumber := p_x509SerialNumber}
                }
            }
        }
    }

    template BodyElement_group mdw_SECC_CMN_CertificateUpdateRes_001(
                               in template (present) CertificateChainType
                               p_sAProvisioningCertificateChain,
                               in template (present) CertificateChainType
                               p_contractSignatureCertChain,
                               in template (present) EMAIDType_1 p_emaid) := {

        certificateUpdateRes := {
            responseCode := oK,
            sAProvisioningCertificateChain := p_sAProvisioningCertificateChain,
            contractSignatureCertChain := p_contractSignatureCertChain,
            contractSignatureEncryptedPrivateKey := ?,
            dHpublickey := ?,
            eMAID := {
                id := ?,
                base := p_emaid
            },
            retryCounter := *
        }
    }

    template BodyElement_group mdw_SECC_CMN_CertificateUpdateRes_002(
                               template (present) ResponseCodeType p_responseCode) := {

        certificateUpdateRes := {
            responseCode := p_responseCode,
            sAProvisioningCertificateChain := ?,
            contractSignatureCertChain := ?,
            contractSignatureEncryptedPrivateKey := ?,
            dHpublickey := ?,
            eMAID := ?,
            retryCounter := ?
        }
    }
}
```

N.3.2.7 SECC templates for CertificateInstallation

```
module Templates_SECC_CertificateInstallation {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from XSDAUX language "XSD" all;
    import from urn_iso_15118_2_2013_X509Def language "XSD" all;

    template BodyElement_group md_SECC_CMN_CertificateInstallationReq_001(
                               in template(present) XSDAUX.ID p_id,
                               in template(present) X509CertificateType p_oEMProvisioningCert,
                               in template(present) XSDAUX.string p_x509IssuerName,
                               in template(present) XSDAUX.integerXSD p_x509SerialNumber) := {

        certificateInstallationReq := {
            id := p_id,
            oEMProvisioningCert := p_oEMProvisioningCert,
            listOfRootCertificateIDs := {
                rootCertificateID_list := {
                    {
                    x509IssuerName := p_x509IssuerName,
                    x509SerialNumber := p_x509SerialNumber
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    }
                }
            }
        }
    }

    template BodyElement_group mdw_SECC_CMN_CertificateInstallationRes_001(
                              in template (present) CertificateChainType
                              p_sAProvisioningCertificateChain,
                              in template (present) CertificateChainType
                              p_contractSignatureCertChain) := {

        certificateInstallationRes := {
            responseCode := oK,
            sAProvisioningCertificateChain := p_sAProvisioningCertificateChain,
            contractSignatureCertChain := p_contractSignatureCertChain,
            contractSignatureEncryptedPrivateKey := ?,
            dHpublickey := ?,
            eMAID := ?
        }
    }

    template BodyElement_group mdw_SECC_CMN_CertificateInstallationRes_002(
                              template (present) ResponseCodeType p_responseCode) := {

        certificateInstallationRes := {
            responseCode := p_responseCode,
            sAProvisioningCertificateChain := ?,
            contractSignatureCertChain := ?,
            contractSignatureEncryptedPrivateKey := ?,
            dHpublickey := ?,
            eMAID := ?
        }
    }
}
```

N.3.2.8 SECC templates for MeteringReceipt

```
module Templates_SECC_MeteringReceipt {

   import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
   import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
   import from XSDAUX language "XSD" all;

   template BodyElement_group md_SECC_CMN_MeteringReceiptReq_001(
                                 template XSDAUX.ID p_id,
                              template (present) SessionIDType p_sessionId,
                              template (present) MeterInfoType p_meterInfo,
                              template SAIDType p_sAScheduleTupleId) := {

      meteringReceiptReq :=
       {
        id := p_id,
        sessionID := p_sessionId,
        sAScheduleTupleID := p_sAScheduleTupleId,
        meterInfo := p_meterInfo
       }
   }

   template BodyElement_group md_SECC_AC_MeteringReceiptRes_001(
                                 template (present) ResponseCodeType p_responseCode,
                              template (present) AC_EVSEStatusType  p_aCeVSEStatus) := {

      meteringReceiptRes :=
       {
        responseCode := p_responseCode,
        eVSEStatus :=
         {
          aC_EVSEStatus := p_aCeVSEStatus
         }
       }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
   }

   template BodyElement_group md_SECC_DC_MeteringReceiptRes_001(
                                template (present) ResponseCodeType p_responseCode,
                              template (present) DC_EVSEStatusType  p_dCeVSEStatus) := {

      meteringReceiptRes :=
       {
        responseCode := p_responseCode,
        eVSEStatus :=
         {
          dC_EVSEStatus := p_dCeVSEStatus
         }
       }
   }
 }
```

N.4 EVCC templates

N.4.1 V2G protocol handshake

N.4.1.1 EVCC templates for SupportedAppProtocol

```
module Templates_EVCC_SupportedAppProtocol {

    import from urn_iso_15118_2_2010_AppProtocol language "XSD" all;
    import from XSDAUX all;

    template SupportedAppProtocolReq mw_EVCC_CMN_SupportedAppProtocolReq_001() := {
        appProtocol_list := ? length (1 .. 20)
    }
}

with {
    encode "ISO_V2G"
}
```

N.4.2 V2G messages

N.4.2.1 EVCC templates for ServiceDetail

```
module Templates_EVCC_ServiceDetail {

       import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
       import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
       import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
       import from XSDAUX language "XSD" all;

       template BodyElement_group md_EVCC_CMN_ServiceDetailReq_001(
                                   template (present) ServiceIDType p_serviceID) := {
         serviceDetailReq := {
             serviceID := p_serviceID
             }
       }

      template BodyElement_group md_EVCC_CMN_ServiceDetailRes_001(
                                template (present) ResponseCodeType p_responseCode,
                                template (present) ServiceIDType p_serviceID) := {

        serviceDetailRes := {
            responseCode := p_responseCode,
            serviceID := p_serviceID,
            serviceParameterList := {
                parameterSet_list := {
                    {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    parameterSetID := 1,
                    parameter_list := {
                        {name := "Service", choice := {stringValue := "Installation"}}
                    }
                },
                {
                    parameterSetID := 2,
                    parameter_list := {
                        {name := "Service", choice := {stringValue := "Update"}}
                    }
                }
            }
        }
    }
}

template BodyElement_group md_EVCC_CMN_ServiceDetailRes_002(
                            template (present) ResponseCodeType p_responseCode,
                            template (present) ServiceIDType p_serviceID) := {

     serviceDetailRes := {
         responseCode := p_responseCode,
         serviceID := p_serviceID,
         serviceParameterList := {
             parameterSet_list := {
                 {
                     parameterSetID := 1,
                     parameter_list := {
                         {name := "Protocol", choice := {stringValue := "ftp"}},
                         {name := "Port", choice := {intValue := 20}}
                     }
                 },
                 {
                     parameterSetID := 2,
                     parameter_list := {
                         {name := "Protocol", choice := {stringValue := "ftp"}},
                         {name := "Port", choice := {intValue := 21}}
                     }
                 },
                 {
                     parameterSetID := 3,
                     parameter_list := {
                         {name := "Protocol", choice := {stringValue := "http"}},
                         {name := "Port", choice := {intValue := 80}}
                     }
                 },
                 {
                     parameterSetID := 4,
                     parameter_list := {
                         {name := "Protocol", choice := {stringValue := "https"}},
                         {name := "Port", choice := {intValue := 443}}
                     }
                 }
             }
         }
     }
 }

template BodyElement_group md_EVCC_CMN_ServiceDetailRes_003(
                            template (present) ResponseCodeType p_responseCode,
                            template (present) ServiceIDType p_serviceID,
                            template (present) XSDAUX.short p_parameterSetID,
                            template (present) XSDAUX.string p_protocol,
                            template (present) XSDAUX.int p_port) := {

     serviceDetailRes := {
         responseCode := p_responseCode,
         serviceID := p_serviceID,
         serviceParameterList := {
             parameterSet_list := {
                 {
                     parameterSetID := p_parameterSetID,
                     parameter_list := {
                         {name := "Protocol", choice := {stringValue := p_protocol}},
                         {name := "Port", choice := {intValue := p_port}}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                }
                            }
                        }
                    }
                }
            }
        }
}
```

N.4.2.2 EVCC templates for PaymentServiceSelection

```
module Templates_EVCC_PaymentServiceSelection {

   import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
   import from urn_iso_15118_2_2013_MsgBody language "XSD" all;

   template BodyElement_group md_EVCC_CMN_PaymentServiceSelectionReq_001(
                              template (present) PaymentOptionType p_selPaymentOption,
                              template (present) SelectedServiceListType p_selectedServiceList) := {

      paymentServiceSelectionReq := {
        selectedPaymentOption := p_selPaymentOption,
        selectedServiceList   := p_selectedServiceList
       }
    }

    template SelectedServiceListType m_EVCC_CMN_SelectedServiceList_001(
                                     template (present) ServiceIDType p_serviceID) := {

         selectedService_list := {
             {serviceID := p_serviceID, parameterSetID := *}
         }
    }

    template BodyElement_group md_EVCC_CMN_PaymentServiceSelectionRes_001(
                               template (present) ResponseCodeType p_responseCode) := {

         paymentServiceSelectionRes := {
             responseCode := p_responseCode
       }
    }
 }
```

N.4.2.3 EVCC templates for PaymentDetails

```
module Templates_EVCC_PaymentDetails {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from XSDAUX language "XSD" all;

    template BodyElement_group md_EVCC_CMN_PaymentDetailsReq_001(
                               template (present) EMAIDType_1 p_eMAID,
                               template (present) CertificateChainType
                               p_contractSignatureCertChain) := {

        paymentDetailsReq := {
            eMAID := p_eMAID,
            contractSignatureCertChain := p_contractSignatureCertChain
        }
    }

    template BodyElement_group md_EVCC_CMN_PaymentDetailsRes_001(
                               template(present) ResponseCodeType p_responseCode,
                               template(present) GenChallengeType p_genChallenge,
                               template(present) XSDAUX.long p_eVSETimeStamp) := {

        paymentDetailsRes := {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            responseCode := p_responseCode,
            genChallenge := p_genChallenge,
            eVSETimeStamp := p_eVSETimeStamp
        }
    }
}
```

N.4.2.4 EVCC templates for ChargeParameterDiscovery

```
module Templates_EVCC_ChargeParameterDiscovery {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from XSDAUX language "XSD" all;
    import from Timer_15118 all;
    import from DataStructure_V2G all;

    template BodyElement_group md_EVCC_CMN_ChargeParamterDiscoveryReq_001(
                               template XSDAUX.unsignedShort p_maxEntriesSAScheduleTuple,
                               template(present) EnergyTransferModeType p_requestedEnergyTransferMode,
                               template(present) EVChargeParameter_group p_eVChargeParameter) := {

        chargeParameterDiscoveryReq := {
            maxEntriesSAScheduleTuple := p_maxEntriesSAScheduleTuple,
            requestedEnergyTransferMode := p_requestedEnergyTransferMode,
            eVChargeParameter := p_eVChargeParameter
        }
    }

    template EVChargeParameter_group mw_EVCC_AC_EVChargeParameter_001() := {
        aC_EVChargeParameter := {
            departureTime := omit,
            eAmount := ?,
            eVMaxVoltage := ?,
            eVMaxCurrent := ?,
            eVMinCurrent := ?
        }
    }

    template EVChargeParameter_group mw_EVCC_DC_EVChargeParameter_001() := {
        dC_EVChargeParameter := {
            departureTime := omit,
            dC_EVStatus := ?,
            eVMaximumCurrentLimit := ?,
            eVMaximumPowerLimit := *,
            eVMaximumVoltageLimit := ?,
            eVEnergyCapacity := *,
            eVEnergyRequest := *,
            fullSOC := *,
            bulkSOC := *
        }
    }

    template BodyElement_group md_EVCC_CMN_ChargeParameterDiscoveryRes_001(
                               template(present) ResponseCodeType p_responseCode,
                               template(present) EVSEProcessingType p_eVSEProcessing,
                               template SASchedules_group p_sASchedules,
                               template(present) EVSEChargeParameter_group
                               p_eVSEChargeParameter) := {

        chargeParameterDiscoveryRes := {
            responseCode := p_responseCode,
            eVSEProcessing := p_eVSEProcessing,
            sASchedules := p_sASchedules,
            eVSEChargeParameter := p_eVSEChargeParameter
        }
    }

    template BodyElement_group md_EVCC_CMN_ChargeParameterDiscoveryRes_002(
                               template(present) ResponseCodeType p_responseCode,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                            template(present) EVSEProcessingType p_eVSEProcessing,
                            template(present) EVSEChargeParameter_group
                            p_eVSEChargeParameter) := {

    chargeParameterDiscoveryRes := {
        responseCode := p_responseCode,
        eVSEProcessing := p_eVSEProcessing,
        sASchedules := omit,
        eVSEChargeParameter := p_eVSEChargeParameter
    }
}

template SASchedules_group md_EVCC_CMN_SASchedules_001(
                            template(present) StartList v_start,
                            template DurationType v_duration,
                            template (present) SAIDType p_sAScheduleTupleID1,
                            template (present) SAIDType p_sAScheduleTupleID2,
                            template (present) SAIDType p_sAScheduleTupleID3,
                            template SalesTariffType v_salesTariff1,
                            template SalesTariffType v_salesTariff2,
                            template SalesTariffType v_salesTariff3,
                            template PmaxList v_pMax) := {

    sAScheduleList := {
        sAScheduleTuple_list := {
            {
                sAScheduleTupleID := p_sAScheduleTupleID1,
                pMaxSchedule := {
                    pMaxScheduleEntry_list := {
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[0], duration := omit
                                }
                            },
                            pMax := v_pMax[0]
                        },
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[1], duration := omit
                                }
                            },
                            pMax := v_pMax[1]
                        },
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[2], duration := omit
                                }
                            },
                            pMax := v_pMax[2]
                        },
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[3], duration := omit
                                }
                            },
                            pMax := v_pMax[3]
                        },
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[4], duration := omit
                                }
                            },
                            pMax := v_pMax[4]
                        },
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[5], duration := omit
                                }
                            },
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                    pMax := v_pMax[5]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[6], duration := omit
                        }
                    },
                    pMax := v_pMax[6]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[7], duration := omit
                        }
                    },
                    pMax := v_pMax[7]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[8], duration := omit
                        }
                    },
                    pMax := v_pMax[8]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[9], duration := omit
                        }
                    },
                    pMax := v_pMax[9]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[10], duration := omit
                        }
                    },
                    pMax := v_pMax[10]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[11], duration := v_duration
                        }
                    },
                    pMax := v_pMax[11]
                }
            }
        },
        salesTariff := v_salesTariff1
    },
    {
        sAScheduleTupleID := p_sAScheduleTupleID2,
        pMaxSchedule := {
            pMaxScheduleEntry_list := {
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[0], duration := omit
                        }
                    },
                    pMax := v_pMax[0]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[1], duration := omit
                        }
                    },
                    pMax := v_pMax[1]
                },
                {
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[2], duration := omit
            }
        },
        pMax := v_pMax[2]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[3], duration := omit
            }
        },
        pMax := v_pMax[3]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[4], duration := omit
            }
        },
        pMax := v_pMax[4]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[5], duration := omit
            }
        },
        pMax := v_pMax[5]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[6], duration := omit
            }
        },
        pMax := v_pMax[6]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[7], duration := omit
            }
        },
        pMax := v_pMax[7]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[8], duration := omit
            }
        },
        pMax := v_pMax[8]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[9], duration := omit
            }
        },
        pMax := v_pMax[9]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[10], duration := omit
            }
        },
        pMax := v_pMax[10]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[11], duration := v_duration
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        }
                    },
                    pMax := v_pMax[11]
                }
            }
        },
        salesTariff := v_salesTariff2
    },
    {
        sAScheduleTupleID := p_sAScheduleTupleID3,
        pMaxSchedule := {
            pMaxScheduleEntry_list := {
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[0], duration := omit
                        }
                    },
                    pMax := v_pMax[0]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[1], duration := omit
                        }
                    },
                    pMax := v_pMax[1]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[2], duration := omit
                        }
                    },
                    pMax := v_pMax[2]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[3], duration := omit
                        }
                    },
                    pMax := v_pMax[3]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[4], duration := omit
                        }
                    },
                    pMax := v_pMax[4]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[5], duration := omit
                        }
                    },
                    pMax := v_pMax[5]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[6], duration := omit
                        }
                    },
                    pMax := v_pMax[6]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[7], duration := omit
                        }
                    },
                    pMax := v_pMax[7]
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[8], duration := omit
                                    }
                                },
                                pMax := v_pMax[8]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[9], duration := omit
                                    }
                                },
                                pMax := v_pMax[9]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[10], duration := omit
                                    }
                                },
                                pMax := v_pMax[10]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[11], duration := v_duration
                                    }
                                },
                                pMax := v_pMax[11]
                            }
                        }
                    },
                    salesTariff := v_salesTariff3
                }
            }
        }
    }

    template SASchedules_group md_EVCC_CMN_SASchedules_002(
                                template(present) StartList v_start,
                                template(present) StartType v_start0W,
                                template DurationType v_duration,
                                template (present) SAIDType p_sAScheduleTupleID1,
                                template (present) SAIDType p_sAScheduleTupleID2,
                                template (present) SAIDType p_sAScheduleTupleID3,
                                template SalesTariffType v_salesTariff1,
                                template SalesTariffType v_salesTariff2,
                                template SalesTariffType v_salesTariff3,
                                template PmaxList v_pMax) := {

        sAScheduleList := {
            sAScheduleTuple_list := {
                {
                    sAScheduleTupleID := p_sAScheduleTupleID1,
                    pMaxSchedule := {
                        pMaxScheduleEntry_list := {
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[0], duration := omit
                                    }
                                },
                                pMax := {
                                    multiplier := 0, unit := w, value_ := 0
                                }
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start0W,
                                        duration := omit
                                    }
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        },
        pMax := v_pMax[1]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[2], duration := omit
            }
        },
        pMax := v_pMax[2]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[3], duration := omit
            }
        },
        pMax := v_pMax[3]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[4], duration := omit
            }
        },
        pMax := v_pMax[4]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[5], duration := omit
            }
        },
        pMax := v_pMax[5]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[6], duration := omit
            }
        },
        pMax := v_pMax[6]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[7], duration := omit
            }
        },
        pMax := v_pMax[7]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[8], duration := omit
            }
        },
        pMax := v_pMax[8]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[9], duration := omit
            }
        },
        pMax := v_pMax[9]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[10], duration := omit
            }
        },
        pMax := v_pMax[10]
    },
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[11], duration := v_duration
                        }
                    },
                    pMax := v_pMax[11]
                }
            }
        },
        salesTariff := v_salesTariff1
    },
    {
        sAScheduleTupleID := p_sAScheduleTupleID2,
        pMaxSchedule := {
            pMaxScheduleEntry_list := {
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[0], duration := omit
                        }
                    },
                    pMax := {
                        multiplier := 0, unit := w, value_ := 0
                    }
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start0W,
                            duration := omit
                        }
                    },
                    pMax := v_pMax[1]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[2], duration := omit
                        }
                    },
                    pMax := v_pMax[2]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[3], duration := omit
                        }
                    },
                    pMax := v_pMax[3]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[4], duration := omit
                        }
                    },
                    pMax := v_pMax[4]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[5], duration := omit
                        }
                    },
                    pMax := v_pMax[5]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[6], duration := omit
                        }
                    },
                    pMax := v_pMax[6]
                },
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[7], duration := omit
                        }
                    },
                    pMax := v_pMax[7]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[8], duration := omit
                        }
                    },
                    pMax := v_pMax[8]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[9], duration := omit
                        }
                    },
                    pMax := v_pMax[9]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[10], duration := omit
                        }
                    },
                    pMax := v_pMax[10]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[11], duration := v_duration
                        }
                    },
                    pMax := v_pMax[11]
                }
            }
        },
        salesTariff := v_salesTariff2
    },
    {
        sAScheduleTupleID := p_sAScheduleTupleID3,
        pMaxSchedule := {
            pMaxScheduleEntry_list := {
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[0], duration := omit
                        }
                    },
                    pMax := {
                        multiplier := 0, unit := w, value_ := 0
                    }
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start0W,
                            duration := omit
                        }
                    },
                    pMax := v_pMax[1]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[2], duration := omit
                        }
                    },
                    pMax := v_pMax[2]
                },
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[3], duration := omit
                        }
                    },
                    pMax := v_pMax[3]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[4], duration := omit
                        }
                    },
                    pMax := v_pMax[4]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[5], duration := omit
                        }
                    },
                    pMax := v_pMax[5]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[6], duration := omit
                        }
                    },
                    pMax := v_pMax[6]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[7], duration := omit
                        }
                    },
                    pMax := v_pMax[7]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[8], duration := omit
                        }
                    },
                    pMax := v_pMax[8]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[9], duration := omit
                        }
                    },
                    pMax := v_pMax[9]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[10], duration := omit
                        }
                    },
                    pMax := v_pMax[10]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[11], duration := v_duration
                        }
                    },
                    pMax := v_pMax[11]
                }
            }
        },
        salesTariff := v_salesTariff3
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                }
            }
        }
    }

    template SASchedules_group md_EVCC_CMN_SASchedules_003(
                                template(present) StartList v_start,
                                template(present) StartType v_start0W,
                                template DurationType v_duration,
                                template (present) SAIDType p_sAScheduleTupleID1,
                                template (present) SAIDType p_sAScheduleTupleID2,
                                template (present) SAIDType p_sAScheduleTupleID3,
                                template SalesTariffType v_salesTariff1,
                                template SalesTariffType v_salesTariff2,
                                template SalesTariffType v_salesTariff3,
                                template PmaxList v_pMax) := {

        sAScheduleList := {
            sAScheduleTuple_list := {
                {
                    sAScheduleTupleID := p_sAScheduleTupleID1,
                    pMaxSchedule := {
                        pMaxScheduleEntry_list := {
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[0], duration := omit
                                    }
                                },
                                pMax := v_pMax[0]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[1], duration := omit
                                    }
                                },
                                pMax := {
                                    multiplier := 0, unit := w, value_ := 0
                                }
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start0W, duration := omit
                                    }
                                },
                                pMax := v_pMax[2]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[3], duration := omit
                                    }
                                },
                                pMax := v_pMax[3]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[4], duration := omit
                                    }
                                },
                                pMax := v_pMax[4]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[5], duration := omit
                                    }
                                },
                                pMax := v_pMax[5]
                            },
                            {
                                timeInterval := {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        relativeTimeInterval := {
                            start_ := v_start[6], duration := omit
                        }
                    },
                    pMax := v_pMax[6]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[7], duration := omit
                        }
                    },
                    pMax := v_pMax[7]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[8], duration := omit
                        }
                    },
                    pMax := v_pMax[8]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[9], duration := omit
                        }
                    },
                    pMax := v_pMax[9]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[10], duration := omit
                        }
                    },
                    pMax := v_pMax[10]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[11], duration := v_duration
                        }
                    },
                    pMax := v_pMax[11]
                }
            }
        },
        salesTariff := v_salesTariff1
    },
    {
        sAScheduleTupleID := p_sAScheduleTupleID2,
        pMaxSchedule := {
            pMaxScheduleEntry_list := {
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[0], duration := omit
                        }
                    },
                    pMax := v_pMax[0]
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
                            start_ := v_start[1], duration := omit
                        }
                    },
                    pMax := {
                        multiplier := 0, unit := w, value_ := 0
                    }
                },
                {
                    timeInterval := {
                        relativeTimeInterval := {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                start_ := v_start0W, duration := omit
            }
        },
        pMax := v_pMax[2]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[3], duration := omit
            }
        },
        pMax := v_pMax[3]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[4], duration := omit
            }
        },
        pMax := v_pMax[4]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[5], duration := omit
            }
        },
        pMax := v_pMax[5]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[6], duration := omit
            }
        },
        pMax := v_pMax[6]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[7], duration := omit
            }
        },
        pMax := v_pMax[7]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[8], duration := omit
            }
        },
        pMax := v_pMax[8]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[9], duration := omit
            }
        },
        pMax := v_pMax[9]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[10], duration := omit
            }
        },
        pMax := v_pMax[10]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[11], duration := v_duration
            }
        },
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        pMax := v_pMax[11]
                    }
                }
            },
            salesTariff := v_salesTariff2
        },
        {
            sAScheduleTupleID := p_sAScheduleTupleID3,
            pMaxSchedule := {
                pMaxScheduleEntry_list := {
                    {
                        timeInterval := {
                            relativeTimeInterval := {
                                start_ := v_start[0], duration := omit
                            }
                        },
                        pMax := v_pMax[0]
                    },
                    {
                        timeInterval := {
                            relativeTimeInterval := {
                                start_ := v_start[1], duration := omit
                            }
                        },
                        pMax := {
                            multiplier := 0, unit := w, value_ := 0
                        }
                    },
                    {
                        timeInterval := {
                            relativeTimeInterval := {
                                start_ := v_start0W, duration := omit
                            }
                        },
                        pMax := v_pMax[2]
                    },
                    {
                        timeInterval := {
                            relativeTimeInterval := {
                                start_ := v_start[3], duration := omit
                            }
                        },
                        pMax := v_pMax[3]
                    },
                    {
                        timeInterval := {
                            relativeTimeInterval := {
                                start_ := v_start[4], duration := omit
                            }
                        },
                        pMax := v_pMax[4]
                    },
                    {
                        timeInterval := {
                            relativeTimeInterval := {
                                start_ := v_start[5], duration := omit
                            }
                        },
                        pMax := v_pMax[5]
                    },
                    {
                        timeInterval := {
                            relativeTimeInterval := {
                                start_ := v_start[6], duration := omit
                            }
                        },
                        pMax := v_pMax[6]
                    },
                    {
                        timeInterval := {
                            relativeTimeInterval := {
                                start_ := v_start[7], duration := omit
                            }
                        },
                        pMax := v_pMax[7]
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[8], duration := omit
                                    }
                                },
                                pMax := v_pMax[8]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[9], duration := omit
                                    }
                                },
                                pMax := v_pMax[9]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[10], duration := omit
                                    }
                                },
                                pMax := v_pMax[10]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[11], duration := v_duration
                                    }
                                },
                                pMax := v_pMax[11]
                            }
                        }
                    },
                    salesTariff := v_salesTariff3
                }
            }
        }
    }

    template SASchedules_group md_EVCC_CMN_SASchedules_004(
                                template(present) SAIDType p_sAScheduleTupleID,
                                template(present) StartList v_start,
                                template DurationType v_duration,
                                template PmaxList v_pMax) := {

        sAScheduleList := {
            sAScheduleTuple_list := {
                {
                    sAScheduleTupleID := p_sAScheduleTupleID,
                    pMaxSchedule := {
                        pMaxScheduleEntry_list := {
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[0], duration := omit
                                    }
                                },
                                pMax := v_pMax[0]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[1], duration := omit
                                    }
                                },
                                pMax := v_pMax[1]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[2], duration := omit
                                    }
                                },
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                pMax := v_pMax[2]
            },
            {
                timeInterval := {
                    relativeTimeInterval := {
                        start_ := v_start[3], duration := omit
                    }
                },
                pMax := v_pMax[3]
            },
            {
                timeInterval := {
                    relativeTimeInterval := {
                        start_ := v_start[4], duration := omit
                    }
                },
                pMax := v_pMax[4]
            },
            {
                timeInterval := {
                    relativeTimeInterval := {
                        start_ := v_start[5], duration := omit
                    }
                },
                pMax := v_pMax[5]
            },
            {
                timeInterval := {
                    relativeTimeInterval := {
                        start_ := v_start[6], duration := omit
                    }
                },
                pMax := v_pMax[6]
            },
            {
                timeInterval := {
                    relativeTimeInterval := {
                        start_ := v_start[7], duration := omit
                    }
                },
                pMax := v_pMax[7]
            },
            {
                timeInterval := {
                    relativeTimeInterval := {
                        start_ := v_start[8], duration := omit
                    }
                },
                pMax := v_pMax[8]
            },
            {
                timeInterval := {
                    relativeTimeInterval := {
                        start_ := v_start[9], duration := omit
                    }
                },
                pMax := v_pMax[9]
            },
            {
                timeInterval := {
                    relativeTimeInterval := {
                        start_ := v_start[10], duration := omit
                    }
                },
                pMax := v_pMax[10]
            },
            {
                timeInterval := {
                    relativeTimeInterval := {
                        start_ := v_start[11], duration := v_duration
                    }
                },
                pMax := v_pMax[11]
            }
        }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                    },
                    salesTariff := omit
                }
            }
        }
    }

template SASchedules_group md_EVCC_CMN_SASchedules_005(
                            template(present) SAIDType p_sAScheduleTupleID,
                            template(present) StartList v_start,
                            template(present) StartType v_start0W,
                            template DurationType v_duration,
                            template PmaxList v_pMax) := {

    sAScheduleList := {
        sAScheduleTuple_list := {
            {
                sAScheduleTupleID := p_sAScheduleTupleID,
                pMaxSchedule := {
                    pMaxScheduleEntry_list := {
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[0], duration := omit
                                }
                            },
                            pMax := {
                                multiplier := 0, unit := w, value_ := 0
                            }
                        },
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start0W, duration := omit
                                }
                            },
                            pMax := v_pMax[1]
                        },
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[2], duration := omit
                                }
                            },
                            pMax := v_pMax[2]
                        },
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[3], duration := omit
                                }
                            },
                            pMax := v_pMax[3]
                        },
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[4], duration := omit
                                }
                            },
                            pMax := v_pMax[4]
                        },
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[5], duration := omit
                                }
                            },
                            pMax := v_pMax[5]
                        },
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := v_start[6], duration := omit
                                }
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                },
                                pMax := v_pMax[6]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[7], duration := omit
                                    }
                                },
                                pMax := v_pMax[7]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[8], duration := omit
                                    }
                                },
                                pMax := v_pMax[8]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[9], duration := omit
                                    }
                                },
                                pMax := v_pMax[9]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[10], duration := omit
                                    }
                                },
                                pMax := v_pMax[10]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[11], duration := v_duration
                                    }
                                },
                                pMax := v_pMax[11]
                            }
                        }
                    },
                    salesTariff := omit
                }
            }
        }
    }

template SASchedules_group md_EVCC_CMN_SASchedules_006(
                            template(present) SAIDType p_sAScheduleTupleID,
                            template(present) StartList v_start,
                            template(present) StartType v_start0W,
                            template DurationType v_duration,
                            template PmaxList v_pMax) := {

    sAScheduleList := {
        sAScheduleTuple_list := {
            {
                sAScheduleTupleID := p_sAScheduleTupleID,
                pMaxSchedule := {
                    pMaxScheduleEntry_list := {
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[0], duration := omit
                                    }
                                },
                                pMax := v_pMax[0]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                start_ := v_start[1], duration := omit
            }
        },
        pMax := {
            multiplier := 0, unit := w, value_ := 0
        }
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start0W, duration := omit
            }
        },
        pMax := v_pMax[2]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[3], duration := omit
            }
        },
        pMax := v_pMax[3]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[4], duration := omit
            }
        },
        pMax := v_pMax[4]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[5], duration := omit
            }
        },
        pMax := v_pMax[5]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[6], duration := omit
            }
        },
        pMax := v_pMax[6]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[7], duration := omit
            }
        },
        pMax := v_pMax[7]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[8], duration := omit
            }
        },
        pMax := v_pMax[8]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[9], duration := omit
            }
        },
        pMax := v_pMax[9]
    },
    {
        timeInterval := {
            relativeTimeInterval := {
                start_ := v_start[10], duration := omit
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                    }
                                },
                                pMax := v_pMax[10]
                            },
                            {
                                timeInterval := {
                                    relativeTimeInterval := {
                                        start_ := v_start[11], duration := v_duration
                                    }
                                },
                                pMax := v_pMax[11]
                            }
                        }
                    },
                    salesTariff := omit
                }
            }
        }
    }

template SASchedules_group md_EVCC_CMN_SASchedules_007(
                            template DurationType v_duration,
                            template (present) SAIDType p_sAScheduleTupleID1,
                            template (present) SAIDType p_sAScheduleTupleID2,
                            template (present) SAIDType p_sAScheduleTupleID3,
                            template SalesTariffType v_salesTariff1,
                            template SalesTariffType v_salesTariff2,
                            template SalesTariffType v_salesTariff3,
                            template PhysicalValueType v_pMax) := {

    sAScheduleList := {
        sAScheduleTuple_list := {
            {
                sAScheduleTupleID := p_sAScheduleTupleID1,
                pMaxSchedule := {
                    pMaxScheduleEntry_list := {
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := 0, duration := v_duration
                                }
                            },
                            pMax := v_pMax
                        }
                    }
                },
                salesTariff := v_salesTariff1
            },
            {
                sAScheduleTupleID := p_sAScheduleTupleID2,
                pMaxSchedule := {
                    pMaxScheduleEntry_list := {
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := 0, duration := v_duration
                                }
                            },
                            pMax := v_pMax
                        }
                    }
                },
                salesTariff := v_salesTariff2
            },
            {
                sAScheduleTupleID := p_sAScheduleTupleID3,
                pMaxSchedule := {
                    pMaxScheduleEntry_list := {
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := 0, duration := v_duration
                                }
                            },
                            pMax := v_pMax
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                        }
                    }
                },
                salesTariff := v_salesTariff3
            }
        }
    }
}

template SASchedules_group md_EVCC_CMN_SASchedules_008(
                            template(present) SAIDType p_sAScheduleTupleID,
                            template DurationType v_duration,
                            template PhysicalValueType v_pMax) := {

    sAScheduleList := {
        sAScheduleTuple_list := {
            {
                sAScheduleTupleID := p_sAScheduleTupleID,
                pMaxSchedule := {
                    pMaxScheduleEntry_list := {
                        {
                            timeInterval := {
                                relativeTimeInterval := {
                                    start_ := 0,
                                    duration := v_duration
                                }
                            },
                            pMax := v_pMax
                        }
                    }
                },
                salesTariff := omit
            }
        }
    }
 }

template SalesTariffType md_EVCC_CMN_Salestariff_001(template XSDAUX.ID v_id,
                                                     template(present) StartList v_start,
                                                     template DurationType v_duration,
                                                     template(present)SAIDType v_salesTariffID,
                                                     template(present)XSDAUX.unsignedByte
                                                     v_ePriceLevel) := {

    id := v_id,
    salesTariffID := v_salesTariffID,
    salesTariffDescription := omit,
    numEPriceLevels := 3,
    salesTariffEntry_list := {
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := v_start[0], duration := omit
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        },
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := v_start[1], duration := omit
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        },
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := v_start[2], duration := omit
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        },
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := v_start[3], duration := omit
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        },
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := v_start[4], duration := omit
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        },
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := v_start[5], duration := omit
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        },
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := v_start[6], duration := omit
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        },
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := v_start[7], duration := omit
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        },
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := v_start[8], duration := omit
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        },
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := v_start[9], duration := omit
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        },
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := v_start[10], duration := omit
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        },
        {
            timeInterval := {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                relativeTimeInterval := {
                    start_ := v_start[11],
                    duration := v_duration
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        }
    }
 }

template SalesTariffType md_EVCC_CMN_Salestariff_002(template XSDAUX.ID v_id,
                                                     template DurationType v_duration,
                                                     template(present)SAIDType v_salesTariffID,
                                                     template(present)XSDAUX.unsignedByte
                                                     v_ePriceLevel) := {

    id := v_id,
    salesTariffID := v_salesTariffID,
    salesTariffDescription := omit,
    numEPriceLevels := 3,
    salesTariffEntry_list := {
        {
            timeInterval := {
                relativeTimeInterval := {
                    start_ := 0,
                    duration := v_duration
                }
            },
            ePriceLevel := v_ePriceLevel,
            consumptionCost_list := {}
        }
    }
 }

  template EVSEChargeParameter_group md_EVCC_DC_EVSEChargeParameter_002(
                                        template(present) DC_EVSEStatusType v_DC_EVSEStatus,
                                      template(present) PhysicalValueType v_maxCurrentLimit,
                                      template(present) PhysicalValueType v_maxPowerLimit,
                                      template(present) PhysicalValueType v_maxVoltageLimit,
                                      template(present) PhysicalValueType v_minCurrentLimmit,
                                      template(present) PhysicalValueType v_minVoltageLimit,
                                      template PhysicalValueType v_eVSECurrentRegulationTolerance,
                                      template(present) PhysicalValueType v_eVSEPeakCurrentRipple,
                                      template PhysicalValueType v_eVSEEnergyToBeDelivered) := {

    dC_EVSEChargeParameter := {
        dC_EVSEStatus := v_DC_EVSEStatus,
        eVSEMaximumCurrentLimit := v_maxCurrentLimit ,
        eVSEMaximumPowerLimit := v_maxPowerLimit,
        eVSEMaximumVoltageLimit    := v_maxVoltageLimit,
        eVSEMinimumCurrentLimit := v_minCurrentLimmit ,
        eVSEMinimumVoltageLimit := v_minVoltageLimit,
        eVSECurrentRegulationTolerance := v_eVSECurrentRegulationTolerance,
        eVSEPeakCurrentRipple := v_eVSEPeakCurrentRipple,
        eVSEEnergyToBeDelivered := v_eVSEEnergyToBeDelivered
    }
}

template EVSEChargeParameter_group m_EVCC_DC_EVSEChargeParameter_001() := {

    dC_EVSEChargeParameter := {
        dC_EVSEStatus := {
            notificationMaxDelay := 2,
            eVSENotification := none_,
            eVSEIsolationStatus := valid,
            eVSEStatusCode := eVSE_Ready
        },
        eVSEMaximumCurrentLimit := {
            multiplier := 0,
            unit := a,
            value_ := 200
        },
        eVSEMaximumPowerLimit := {
            multiplier := 2,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
                unit := w,
                value_ := 50
            },
            eVSEMaximumVoltageLimit := {
                multiplier := 0,
                unit := v,
                value_ := 500
            },
            eVSEMinimumCurrentLimit := {
                multiplier := 0,
                unit := a,
                value_ := 10
            },
            eVSEMinimumVoltageLimit := {
                multiplier := 0,
                unit := v,
                value_ := 400
            },
            eVSECurrentRegulationTolerance := {
                multiplier := 0,
                unit := a,
                value_ := 1
            },
            eVSEPeakCurrentRipple := {
                multiplier := -1,
                unit := a,
                value_ := 15
            },
            eVSEEnergyToBeDelivered := {
                multiplier := 0,
                unit := wh,
                value_ := 10000
            }
        }
    }

    template EVSEChargeParameter_group m_EVCC_AC_EVSEChargeParameter_001(
                                in template(present) PhysicalValueType
                                v_eVSENominalVoltage,
                                in template(present) PhysicalValueType
                                v_eVSEMaxCurrent) := {

        aC_EVSEChargeParameter := {
            aC_EVSEStatus := {
                rCD := false,
                notificationMaxDelay := 2,
                eVSENotification := none_
            },
            eVSENominalVoltage :=v_eVSENominalVoltage,
            eVSEMaxCurrent := v_eVSEMaxCurrent
        }
    }
}
```

N.4.2.5 EVCC templates for CertificateUpdate

```
module Templates_EVCC_CertificateUpdate {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from XSDAUX language "XSD" all;
    import from urn_iso_15118_2_2013_X509Def language "XSD" all;

    template BodyElement_group mdw_EVCC_CMN_CertificateUpdateReq_001(
                            in template (present) CertificateChainType
                            p_contractSignatureCertChain) := {

        certificateUpdateReq := {
            id := ?,
            contractSignatureCertChain := p_contractSignatureCertChain,
```


Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
            eMAID := ?,
            listOfRootCertificateIDs := ?
        }
    }

    template BodyElement_group md_EVCC_CMN_CertificateUpdateRes_001(
                                template(present)ResponseCodeType p_responseCode,
                                template(present) CertificateChainType
                                p_sAProvisioningCertificateChain,
                                template(present) CertificateChainType
                                p_contractSignatureCertChain,
                                template(present) ContractSignatureEncryptedPrivateKeyType
                                p_contractSignatureEncryptedPrivateKey,
                                template(present) DiffieHellmanPublickeyType p_dHpublickey,
                                template(present) EMAIDType p_eMAID) := {

        certificateUpdateRes := {
            responseCode := p_responseCode,
            sAProvisioningCertificateChain := p_sAProvisioningCertificateChain,
            contractSignatureCertChain := p_contractSignatureCertChain,
            contractSignatureEncryptedPrivateKey := p_contractSignatureEncryptedPrivateKey,
            dHpublickey := p_dHpublickey,
            eMAID := p_eMAID,
            retryCounter := omit
        }
    }
}
```

N.4.2.6 EVCC templates for CertificateInstallation

```
module Templates_EVCC_CertificateInstallation {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" all;
    import from XSDAUX language "XSD" all;
    import from urn_iso_15118_2_2013_X509Def language "XSD" all;

    template BodyElement_group mdw_EVCC_CMN_CertificateInstallationReq_001(
                                in template(present) X509CertificateType p_x509Certificate) := {

        certificateInstallationReq := {
            id := ?,
            oEMProvisioningCert := p_x509Certificate,
            listOfRootCertificateIDs := ?
        }
    }

    template BodyElement_group md_EVCC_CMN_CertificateInstallationRes_001(
                                template(present)ResponseCodeType p_responseCode,
                                template(present) CertificateChainType
                                p_sAProvisioningCertificateChain,
                                template(present) CertificateChainType
                                p_contractSignatureCertChain,
                                template(present) ContractSignatureEncryptedPrivateKeyType
                                p_contractSignatureEncryptedPrivateKey,
                                template(present) DiffieHellmanPublickeyType p_dHpublickey,
                                template(present) EMAIDType p_eMAID) := {

        certificateInstallationRes := {
            responseCode := p_responseCode,
            sAProvisioningCertificateChain := p_sAProvisioningCertificateChain,
            contractSignatureCertChain := p_contractSignatureCertChain,
            contractSignatureEncryptedPrivateKey := p_contractSignatureEncryptedPrivateKey,
            dHpublickey := p_dHpublickey,
            eMAID := p_eMAID
        }
    }
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

N.4.2.7 EVCC templates for MeteringReceipt

```
module Templates_EVCC_MeteringReceipt {

   import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
   import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
   import from XSDAUX language "XSD" all;

   template BodyElement_group md_EVCC_CMN_MeteringReceiptReq_001(
                              template XSDAUX.ID p_id,
                              template (present) SessionIDType p_sessionID,
                              template SAIDType p_sAScheduleTupleID,
                              template (present) MeterInfoType p_meterInfo) := {

      meteringReceiptReq :=
       {
        id                := p_id,
        sessionID         := p_sessionID,
        sAScheduleTupleID := p_sAScheduleTupleID,
        meterInfo :=p_meterInfo
      }
    }

    template BodyElement_group md_EVCC_AC_MeteringReceiptRes_001(
                               template (present) ResponseCodeType p_responseCode,
                               template (present) AC_EVSEStatusType p_aCeVSEStatus) := {

      meteringReceiptRes := {

        responseCode         := p_responseCode,
        eVSEStatus           :=
         {
          aC_EVSEStatus      := p_aCeVSEStatus
         }
       }
     }

     template BodyElement_group md_EVCC_DC_MeteringReceiptRes_001(
                                template (present) ResponseCodeType p_responseCode,
                                template (present) DC_EVSEStatusType p_dCeVSEStatus) := {

       meteringReceiptRes := {

         responseCode         := p_responseCode,
         eVSEStatus           :=
          {
           dC_EVSEStatus      := p_dCeVSEStatus
          }
        }
     }
 }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Annex O
(normative)

Template specifications for Security

O.1 General information

This annex covers all Security *templates* in the ATS.

O.2 Common templates

```
module Templates_CMN_Security {

    import from urn_iso_15118_2_2013_MsgDef language "XSD" {type V2G_Message;}
    import from http_www_w3_org_2000_09_xmldsig all;
    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_MsgBody language "XSD" all;
    import from urn_iso_15118_2_2013_MsgHeader all;
    import from XSDAUX language "XSD" {type anyURI, base64Binary;}
    import from DataStructure_Security all;
    import from Services_Security all;
    import from urn_iso_15118_2_2013_X509Def language "XSD" all;
    import from Pics_15118_2 all;
    import from TTlibrary_Logging all;

    template SignedInfo md_CMN_SignedInfoType_001(template(present) charstring p_id,
                                                  template(present) XSDAUX.base64Binary
                                                  p_digestValue) := {
            id := omit,
            canonicalizationMethod := {
              embed_values := omit,
              algorithm := "http://www.w3.org/TR/canonical-exi/",
              elem_list := {}
            },
            signatureMethod := {
              embed_values := omit,
              algorithm := "http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256",
              hMACOutputLength := omit,
              elem_list := {}
            },
            reference_list := {
              {
                id := omit,
                type_ := omit,
                uRI := "#"&p_id,
                transforms := {
                  transform_list := {
                    {
                      embed_values := omit,
                      algorithm := "http://www.w3.org/TR/canonical-exi/",
                      choice_list := {}
                    }
                  }
                },
                digestMethod := {
                  embed_values := omit,
                  algorithm := "http://www.w3.org/2001/04/xmlenc#sha256",
                  elem_list := {}
              },
                digestValue := p_digestValue
            }
        }
    }

    template SignedInfo md_CMN_SignedInfoType_002(in template (value)
                                                  LocalReferenceList
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                                p_referenceList) := {
    id := omit,
    canonicalizationMethod := {
      embed_values := omit,
      algorithm := "http://www.w3.org/TR/canonical-exi/",
      elem_list := {}
    },
    signatureMethod := {
      embed_values := omit,
      algorithm := "http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256",
      hMACOutputLength := omit,
      elem_list := {}
    },
    reference_list := p_referenceList
}

template SignedInfo mw_CMN_SignedInfoType_003() := {
        id := *,
        canonicalizationMethod := {
          embed_values := *,
          algorithm := ?,
          elem_list := ?
        },
        signatureMethod := {
          embed_values := *,
          algorithm := ?,
          hMACOutputLength := *,
          elem_list := ?
        },
        reference_list := ?
}

template http_www_w3_org_2000_09_xmldsig.ReferenceType md_ReferenceType_001(
                                          template(present) charstring p_id,
                                          template(present) XSDAUX.base64Binary
                                          p_digestValue) := {
    id := omit,
      type_ := omit,
    uRI := "#"&p_id,
    transforms := {
        transform_list := {
          {
            embed_values := omit,
            algorithm := "http://www.w3.org/TR/canonical-exi/",
            choice_list := {}
          }
        }
      },
    digestMethod := {
        embed_values := omit,
        algorithm := "http://www.w3.org/2001/04/xmlenc#sha256",
        elem_list := {}
    },
      digestValue := p_digestValue
  }


template CertificateChainType md_CertificateChainType_001(
                              in template (present) X509CertificateType
                              p_certificateValue) := {
    id := omit,
    x509Certificate := p_certificateValue,
    subCertificates := omit
}

template CertificateChainType md_CertificateChainType_002(
                              in template (present) X509CertificateType
                              p_certificateName,
                              in template X509CertificateType
                              p_subCertificateName2,
                              in template X509CertificateType
                              p_subCertificateName1) := {
     id := omit,
     x509Certificate := p_certificateName,
     subCertificates := {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        x509Certificate_list := {
         p_subCertificateName2,
         p_subCertificateName1
        }
      }
 }

template CertificateChainType md_CertificateChainType_003(
                                 in template (present) X509CertificateType
                                 p_certificateName,
                                 in template X509CertificateType
                                 p_subCertificateName2,
                                 in template X509CertificateType
                                 p_subCertificateName1) := {
         id := *,
         x509Certificate := p_certificateName,
         subCertificates := {
           x509Certificate_list := {
            p_subCertificateName2,
            p_subCertificateName1
           }
         }
      }

template Signature md_CMN_Signature_001(in template(present) SignedInfo
                                        p_signedInfo,
                                        in template(value) charstring
                                        p_privateKey) := {

    id := omit,
    signedInfo := p_signedInfo,
    signatureValue := {
      id := omit,
      base := fx_convertOctetstring2Base64Binary(
                fx_signSignatureInfo("SHA256withECDSA",
                fx_canonicalExi(encvalue(p_signedInfo)),
                p_privateKey))
    },
    keyInfo := omit,
    object_list := omit
}

template Signature md_CMN_Signature_002(in template(present) SignedInfo
                                        p_signedInfo) := {

    id := *,
    signedInfo := p_signedInfo,
    signatureValue := ?,
    keyInfo := *,
    object_list := *
}

template X509CertificateType mdw_CMN_X509Certificate_001(
                                    in template(present)
                                    BasicConstraintsType
                                    p_basicConstraints,
                                    in template
                                    KeyUsageType p_keyUsage) := {

    tBSCertificate := {
        version := 3,
        serial_Number := ?,
        signature_Algorithm := ?,
        issuer := ?,
        validity := {
            not_Before := ?, not_After := ?
        },
        subject := ?,
        subject_Public_Key := {
            public_Key_Algorithm := ?,
            eC_Public_Key := {
                publicKey := ?,
                eCParam := ?
            }
        },
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        standard_Extensions :=
        {
            basic_Constraints := p_basicConstraints,
            name_Constraints := *,
            policy_Constraints := *,
            key_Usage := p_keyUsage,
            extended_Key_Usage := *,
            subject_Key_Identifier := *,
            authority_Key_Identifier := *,
            subject_Alternative_Name := *,
            issuer_Alternative_Name := *,
            subject_Directory_Attributes := *,
            cRL_Distribution_Points := *,
            inhibit_Any_Policy := *,
            private_Key_Usage_Period := *,
            certificate_Policies := *,
            policy_Mappings := *
        },
        non_Standard_Extensions := *
    },
    raw_Certificate := ?,
    certificate_Signature := {
        signature_Algorithm := ?,
        signature_ := ?
    }
}

template BasicConstraintsType md_CMN_BasicConstraints_001(
                              in template (present) boolean p_cA,
                              in template integer p_pathLen) := {

    cA := p_cA,
    pathLen := p_pathLen
}

template KeyUsageType md_CMN_KeyUsage_001(
                      in template (present) boolean p_digitalSignature,
                      in template (present) boolean p_nonRepudiation,
                      in template (present) boolean p_keyEncipherment,
                      in template (present) boolean p_dataEncipherment,
                      in template (present) boolean p_keyAgreement,
                      in template (present) boolean p_keyCertSign,
                      in template (present) boolean p_cRLSign,
                      in template (present) boolean p_encipherOnly,
                      in template (present) boolean p_decipherOnly) := {

    digitalSignature := p_digitalSignature,
    nonRepudiation := p_nonRepudiation,
    keyEncipherment := p_keyEncipherment,
    dataEncipherment := p_dataEncipherment,
    keyAgreement := p_keyAgreement,
    keyCertSign := p_keyCertSign,
    cRLSign := p_cRLSign,
    encipherOnly := p_encipherOnly,
    decipherOnly := p_decipherOnly
}

template X509CertificateType  m_CMN_oEMProvisioningCertificate_001() :=
                              mdw_CMN_X509Certificate_001(
                              md_CMN_BasicConstraints_001(false, omit),
                              md_CMN_KeyUsage_001(true,false,false,false,
                              true,false,false,false,false));

template CertificateChainType m_CMN_sAProvisioningCertificateChain_001() :=
                              md_CertificateChainType_002(
                              mdw_CMN_X509Certificate_001(
                              md_CMN_BasicConstraints_001(false, omit),
                              md_CMN_KeyUsage_001(true,false,false,false,
                              false,false,false,false,false)),
                              mdw_CMN_X509Certificate_001(
                              md_CMN_BasicConstraints_001(true, 0),
                              md_CMN_KeyUsage_001(false,false,false,false,
                              false,?,?,false,false)),
                              mdw_CMN_X509Certificate_001(
                              md_CMN_BasicConstraints_001(true, 1),
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
                                md_CMN_KeyUsage_001(false,false,false,false,
                                false,?,?,false,false)));

    template CertificateChainType m_CMN_contractSignatureCertChain_001() :=
                                md_CertificateChainType_003(
                                mdw_CMN_X509Certificate_001(
                                md_CMN_BasicConstraints_001(false, omit),
                                md_CMN_KeyUsage_001(true,true,true,false,true,
                                false,false,false,false)),
                                mdw_CMN_X509Certificate_001(
                                md_CMN_BasicConstraints_001(true, 0),
                                md_CMN_KeyUsage_001(true,true,false,false,
                                false,?,?,false,false)),
                                mdw_CMN_X509Certificate_001(
                                md_CMN_BasicConstraints_001(true, 1),
                                md_CMN_KeyUsage_001(false,false,false,false,
                                false,?,?,false,false)));

    template CertificateChainType m_CMN_contractSignatureCertChainWrongOrder_001() :=
                                md_CertificateChainType_003(
                                mdw_CMN_X509Certificate_001(
                                md_CMN_BasicConstraints_001(false, omit), ?),
                                mdw_CMN_X509Certificate_001(
                                md_CMN_BasicConstraints_001(true, 1), ?),
                                mdw_CMN_X509Certificate_001(
                                md_CMN_BasicConstraints_001(true, 0), ?));



    template ResponseCodeType m_CMN_failedResponseCodeList_001() :=
                                    (fAILED, fAILED_NoCertificateAvailable);

    template ResponseCodeType m_CMN_failedResponseCodeList_002() :=
                                         (fAILED, fAILED_ContractCanceled,
                                          fAILED_NoCertificateAvailable);

    template ResponseCodeType m_CMN_failedResponseCodeList_003() :=
                                     (fAILED, fAILED_NoCertificateAvailable,
                                      fAILED_CertificateNotAllowedAtThisEVSE);

    template ResponseCodeType m_CMN_failedResponseCodeList_004() :=
                                    (fAILED_SignatureError, fAILED_ChallengeInvalid);
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Annex P
(normative)

Data type definitions

P.1 General information

This annex covers all *data type definitions* in the ATS independent on the SUT type.

P.2 Data types for PICS

```
module DataStructure_PICS_15118_2 {

    type enumerated PkiType {
        PKI_1,
        PKI_2,
        PKI_3,
        PKI_4,
        PKI_5,
        PKI_6,
        PKI_7
    }

    type integer LoopCounter (30..infinity);


}



module DataStructure_PICS_15118 {

    type enumerated ChargingMode {
        aC,
        dC
    }

    type enumerated IdentificationMode {
        pnC,
        eIM
    }

    type enumerated PlugType {
        type1,
        type2
    }

    type enumerated CableCapabilityACType {
        capability13A,
        capability20A,
        capability32A,
        capability63A,
        capability70A
    }

    type enumerated EIMDone {
        beforePlugin,
        afterPlugin,
        duringSlac,
        v2gAuthorization
    }

    type enumerated ZeroPow {
        sleepWithoutCharge,
        sleepAfterCharge,
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
        none_
    }

    type enumerated WakeUp {
        tsInitiated,
        sutInitiated,
        sutInitiatedZeroPow
    }


}
```

P.3 Data types for PIXIT

```
module DataStructure_PIXIT_15118_2 {

    type enumerated CertificateHandlingEVCC {
         none_,
         certInstall,
         certUpdate,
         unknown
    }

    type enumerated CertificateHandlingSECC {
         none_,
         certInstall,
         certUpdate
    }

    type enumerated VASEVCC {
         none_,
         serviceDetail,
         unknown
    }

    type enumerated VASSECC {
         none_,
         serviceDetail
    }

    type enumerated WeldingDetection {
         none_,
         weldingDetection,
         unknown
    }

    type enumerated TLS {
         none_,
         tls,
         unknown
    }

    type enumerated PkiType {
        PKI_1,
        PKI_2,
        PKI_3,
        PKI_4,
        PKI_5,
        PKI_6,
        PKI_7,
        PKI_Ext
    }

    type enumerated SalesTariff {
         salesTariff,
         unknown
    }

    type enumerated InternetAccess {
         internetAccess,
         unknown
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }

    type enumerated Ongoing {
         ongoing,
         unknown
    }

    type enumerated Renegotiation {
        renegotiation,
        unknown
    }

    type enumerated DepartureTime {
        departureTime,
        unknown,
        none_
    }

    type enumerated MeterInfo {
        meterInfo,
        unknown
    }

    type enumerated MeterReading {
        meterReading,
        unknown
    }

    type enumerated MaxNumberOfPhasesInUse {
        maxNumberOfPhasesInUse,
        unknown
    }

    type enumerated ChargingProfileOpt {
        optimization,
        unknown,
        none_
    }

    type enumerated OldSAScheduleTupleID {
        oldSAScheduleTupleID,
        unknown
    }

    type enumerated SAScheduleTupleIDMR {
        sAScheduleTupleID,
        unknown
    }

    type enumerated PaymentOption {
        externalPayment,
        contract
    }

    type enumerated ReceiptRequired {
        receiptRequired,
        none_,
        unknown
    }
}



module DataStructure_PIXIT_15118 {

    type enumerated Pause {
        pause,
        unknown,
        none_
    }
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

P.4 Data types for V2G TCP/TLS Port Control

```
module DataStructure_V2G_TCP_TLS_Port_Control {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from urn_iso_15118_2_2013_X509Def all;

    type record V2G_TCP_TLS_Port_Control_Internal_MessageReq {
        V2G_TCP_TLS_Port_Control_Command_TYPE command
    }

    type record V2G_TCP_TLS_Port_Control_Internal_MessageRes {
        V2G_TCP_TLS_Port_Control_Command_TYPE command,
        V2G_TCP_TLS_Port_Control_Ack_TYPE ack,
            union{
                V2G_TCP_TLS_Port_Number_TYPE portNumber,
                V2G_TCP_TLS_Port_Status_TYPE portStatus
            } result
    }

    type enumerated V2G_TCP_TLS_Port_Control_Command_TYPE {
                                e_getPort, e_disconnectPort,
                                e_getPortStatus, e_connectPort};

    type enumerated V2G_TCP_TLS_Port_Control_Ack_TYPE {ACK_FAILED,
                                    ACK_SUCCESS, e_unkownCommand};

    type record V2G_TCP_TLS_Port_Number_TYPE {
        integer intValue
    }

    type record V2G_TCP_TLS_Port_Status_TYPE {
        boolean closed,
        charstring reason optional
    }

    type record V2G_TCP_TLS_Parameter {
        hexstring seccIpaddress,
        integer seccPort,
        hexstring security optional,
        V2G_TLS_Parameter_TYPE tlsParameter optional
    }

    type record V2G_TLS_Parameter_TYPE {
        charstring cipherSuite,
        charstring enabledProtocol optional,
        union{
            CertificateChainType certificateChain,
            X509CertificateType certificate
        } result
    }
}with {
  encode "V2G_Port_Control";
}
```

P.5 Data types for V2GTP

```
module DataStructure_V2GTP {

    import from DataStructure_SDP {type SDP_Message;}
    import from urn_iso_15118_2_2013_MsgDef language "XSD" {type V2G_Message;}
    import from urn_iso_15118_2_2010_AppProtocol language "XSD" all;

     //  Define V2GTP data structure
    type record V2GTP_Message {
        V2GTP_header_TYPE v2gtp_header,
        V2GTP_payload_TYPE v2gtp_payload
    }

    type record V2GTP_header_TYPE {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
        ProtocolVersion_TYPE protocolVersion,
        InvProtocolVersion_TYPE invProtocolVersion,
        PayloadType_TYPE payloadType,
        PayloadLength_TYPE payloadLength optional
    }

    type hexstring ProtocolVersion_TYPE length (2);

    type hexstring InvProtocolVersion_TYPE length (2);

    type hexstring PayloadType_TYPE length (4);

    type hexstring PayloadLength_TYPE length (8);

    type record V2GTP_payload_TYPE {
        union {
          SDP_Message sdp_message,
          V2G_Message v2g_message,
          SupportedAppProtocolReq supportedAppProtocolReq_message,
          SupportedAppProtocolRes supportedAppProtocolRes_message
        } content
    }
}
with {
     encode "V2GTP";
   }
```

P.6 Data types for SDP messages

```
module DataStructure_SDP{

    // Define SDP data structure
    type record SDP_Message {
        union {
          SDP_Request_TYPE sdp_Request,
          SDP_Response_TYPE sdp_Response
        } content
    }
    with {
        encode "SDP";
    }

    type record SDP_Request_TYPE {
        Security_TYPE security,
        TransportProtocol_TYPE transportProtocol
    }

    type record SDP_Response_TYPE {
        SECC_IPaddress_TYPE secc_IPaddress,
        SECC_Port_TYPE secc_Port,
        Security_TYPE security,
        TransportProtocol_TYPE transportProtocol
    }

    type hexstring Security_TYPE length (2)
    with {
        encode "V2G_Port_Control";
    }

    type hexstring TransportProtocol_TYPE length (2)
    with {
        encode "V2G_Port_Control";
    }

    type hexstring SECC_IPaddress_TYPE length (32)
    with {
        encode "V2G_Port_Control";
    }

    type record Sender_IPaddress_TYPE {
        SECC_IPaddress_TYPE ipAddress
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    }
    with {
        encode "V2G_Port_Control";
    }

    type integer SECC_Port_TYPE (0..65535);
}
```

P.7 Data types for V2G messages

```
module urn_iso_15118_2_2013_X509Def {
  import from XSDAUX all;

  type record BasicConstraintsType {
    XSDAUX.Boolean cA,
    XSDAUX.int pathLen optional
  }

  type record ECPublicKeyType {
    XSDAUX.string publicKey optional,
    XSDAUX.string eCParam optional
  }

  type record KeyUsageType {
    XSDAUX.Boolean digitalSignature,
    XSDAUX.Boolean nonRepudiation,
    XSDAUX.Boolean keyEncipherment,
    XSDAUX.Boolean dataEncipherment,
    XSDAUX.Boolean keyAgreement,
    XSDAUX.Boolean keyCertSign,
    XSDAUX.Boolean cRLSign,
    XSDAUX.Boolean encipherOnly,
    XSDAUX.Boolean decipherOnly
  }

  type XSDAUX.base64Binary RawCertificateType;

  type record SubjectPublicKeyType {
    XSDAUX.string public_Key_Algorithm optional,
    ECPublicKeyType eC_Public_Key optional
  }

  type record ValidityType {
    XSDAUX.string not_Before optional,
    XSDAUX.string not_After optional
  }

  type record X509CertificateContentsType {
    XSDAUX.int version optional,
    XSDAUX.string serial_Number optional,
    XSDAUX.string signature_Algorithm optional,
    XSDAUX.string issuer optional,
    ValidityType validity optional,
    XSDAUX.string subject optional,
    SubjectPublicKeyType subject_Public_Key optional,
    X509V3ExtensionsType standard_Extensions optional,
    X509NonStandardExtensionsType non_Standard_Extensions optional
  }

  type record X509CertificateSignatureType {
    XSDAUX.string signature_Algorithm optional,
    XSDAUX.string signature_ optional
  }

  type record X509CertificateType {
    X509CertificateContentsType tBSCertificate optional,
    RawCertificateType raw_Certificate optional,
    X509CertificateSignatureType certificate_Signature optional
  }

  type record X509NonStandardExtensionsType {
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    XSDAUX.string netscape_Comment optional,
    XSDAUX.string netscape_Certificate_Type optional,
    XSDAUX.string old_Authority_Key_Identifier optional,
    XSDAUX.string old_Primary_Key_Attributes optional
  }

  type record X509V3ExtensionsType {
    BasicConstraintsType basic_Constraints optional,
    XSDAUX.string name_Constraints optional,
    XSDAUX.string policy_Constraints optional,
    KeyUsageType key_Usage optional,
    XSDAUX.string extended_Key_Usage optional,
    XSDAUX.string subject_Key_Identifier optional,
    XSDAUX.string authority_Key_Identifier optional,
    XSDAUX.string subject_Alternative_Name optional,
    XSDAUX.string issuer_Alternative_Name optional,
    XSDAUX.string subject_Directory_Attributes optional,
    XSDAUX.string cRL_Distribution_Points optional,
    XSDAUX.int inhibit_Any_Policy optional,
    ValidityType private_Key_Usage_Period optional,
    XSDAUX.string certificate_Policies optional,
    XSDAUX.string policy_Mappings optional
  }

}
with {
  encode "V2G_CI_MsgDef_ISO";
}



module urn_iso_15118_2_2013_MsgHeader {
  import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;

  import from http_www_w3_org_2000_09_xmldsig language "XSD" all;

  type record MessageHeaderType {
    urn_iso_15118_2_2013_MsgDataTypes.SessionIDType sessionID,
    urn_iso_15118_2_2013_MsgDataTypes.NotificationType notification optional,
    http_www_w3_org_2000_09_xmldsig.SignatureType signature_ optional
  }

}
with {
  encode "V2G_CI_MsgDef_ISO";
}



module http_www_w3_org_2001_04_xmlenc {
  import from http_www_w3_org_2000_09_xmldsig language "XSD" all;

  import from XSDAUX all;

  type AgreementMethodType AgreementMethod;

  type record AgreementMethodType {
    record of XSDAUX.string embed_values optional,
    XSDAUX.anyURI algorithm,
    XSDAUX.base64Binary kA_Nonce optional,
    record of XSDAUX.AnyType elem_list,
    http_www_w3_org_2000_09_xmldsig.KeyInfoType originatorKeyInfo optional,
    http_www_w3_org_2000_09_xmldsig.KeyInfoType recipientKeyInfo optional
  }

  type CipherDataType CipherData;

  type record CipherDataType {
    union {
      XSDAUX.base64Binary cipherValue,
      CipherReferenceType cipherReference
    } choice
  }
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
type CipherReferenceType CipherReference;

type record CipherReferenceType {
  XSDAUX.anyURI uRI,
  union {
    record length (0 .. 1) of TransformsType transforms_list
  } choice
}

type EncryptedDataType EncryptedData;

type record EncryptedDataType {
  XSDAUX.anyURI encoding optional,
  XSDAUX.ID id optional,
  XSDAUX.string mimeType optional,
  XSDAUX.anyURI type_ optional,
  EncryptionMethodType encryptionMethod optional,
  http_www_w3_org_2000_09_xmldsig.KeyInfoType keyInfo optional,
  CipherDataType cipherData,
  EncryptionPropertiesType encryptionProperties optional
}

type EncryptedKeyType EncryptedKey;

type record EncryptedKeyType {
  XSDAUX.anyURI encoding optional,
  XSDAUX.ID id optional,
  XSDAUX.string mimeType optional,
  XSDAUX.string recipient optional,
  XSDAUX.anyURI type_ optional,
  EncryptionMethodType encryptionMethod optional,
  http_www_w3_org_2000_09_xmldsig.KeyInfoType keyInfo optional,
  CipherDataType cipherData,
  EncryptionPropertiesType encryptionProperties optional,
  ReferenceList referenceList optional,
  XSDAUX.string carriedKeyName optional
}

type record EncryptedType {
  XSDAUX.anyURI encoding optional,
  XSDAUX.ID id optional,
  XSDAUX.string mimeType optional,
  XSDAUX.anyURI type_ optional,
  EncryptionMethodType encryptionMethod optional,
  http_www_w3_org_2000_09_xmldsig.KeyInfoType keyInfo optional,
  CipherDataType cipherData,
  EncryptionPropertiesType encryptionProperties optional
}

type union EncryptedType_derivations {
  @default
  EncryptedType encryptedType,
  EncryptedDataType encryptedDataType,
  EncryptedKeyType encryptedKeyType
}

type record EncryptionMethodType {
  record of XSDAUX.string embed_values optional,
  XSDAUX.anyURI algorithm,
  KeySizeType keySize optional,
  XSDAUX.base64Binary oAEPparams optional,
  record of XSDAUX.AnyType elem_list
}

type EncryptionPropertiesType EncryptionProperties;

type record EncryptionPropertiesType {
  XSDAUX.ID id optional,
  record length (1 .. infinity) of EncryptionPropertyType encryptionProperty_list
}

type EncryptionPropertyType EncryptionProperty;

type record EncryptionPropertyType {
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    record of XSDAUX.string embed_values optional,
    XSDAUX.ID id optional,
    XSDAUX.anyURI target optional,
    XSDAUX.AnyAttributes attr optional,
    record length (1 .. infinity) of union {
      XSDAUX.AnyType elem
    } choice_list
  }

  type XSDAUX.integerXSD KeySizeType;

  type record ReferenceList {
    record length (1 .. infinity) of union {
      ReferenceType dataReference,
      ReferenceType keyReference
    } choice_list
  }

  type record ReferenceType {
    XSDAUX.anyURI uRI,
    record of XSDAUX.AnyType elem_list
  }

  type record TransformsType {
    record length (1 .. infinity) of http_www_w3_org_2000_09_xmldsig.TransformType transform_list
  }

}
with {
  encode "xenc_schema";
}



module urn_iso_15118_2_2013_MsgDataTypes {
  import from urn_iso_15118_2_2013_X509Def language "XSD" all;

  import from http_www_w3_org_2000_09_xmldsig language "XSD" all;

  import from XSDAUX all;

  type AC_EVChargeParameterType AC_EVChargeParameter;

  type record AC_EVChargeParameterType {
    XSDAUX.unsignedInt departureTime optional,
    PhysicalValueType eAmount,
    PhysicalValueType eVMaxVoltage,
    PhysicalValueType eVMaxCurrent,
    PhysicalValueType eVMinCurrent
  }

  type AC_EVSEChargeParameterType AC_EVSEChargeParameter;

  type record AC_EVSEChargeParameterType {
    AC_EVSEStatusType aC_EVSEStatus,
    PhysicalValueType eVSENominalVoltage,
    PhysicalValueType eVSEMaxCurrent
  }

  type AC_EVSEStatusType AC_EVSEStatus;

  type record AC_EVSEStatusType {
    XSDAUX.unsignedShort notificationMaxDelay,
    EVSENotificationType eVSENotification,
    XSDAUX.Boolean rCD
  }

  type record CertificateChainType {
    XSDAUX.ID id optional,
    urn_iso_15118_2_2013_X509Def.X509CertificateType x509Certificate,
    SubCertificatesType subCertificates optional
  }

  type enumerated ChargeProgressType {
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
  start_,
  stop_,
  renegotiate
}

type record ChargeServiceType {
  ServiceIDType serviceID,
  ServiceNameType serviceName optional,
  ServiceCategoryType serviceCategory,
  ServiceScopeType serviceScope optional,
  XSDAUX.Boolean freeService,
  SupportedEnergyTransferModeType supportedEnergyTransferMode
}

type record ChargingProfileType {
  record length (1 .. 24) of ProfileEntryType profileEntry_list
}

type enumerated ChargingSessionType {
  terminate,
  pause
}

type record ConsumptionCostType {
  PhysicalValueType startValue,
  record length (1 .. 3) of CostType cost_list
}

type record ContractSignatureEncryptedPrivateKeyType {
  XSDAUX.ID id,
  PrivateKeyType base
}

type enumerated CostKindType {
  relativePricePercentage,
  renewableGenerationPercentage,
  carbonDioxideEmission
}

type record CostType {
  CostKindType costKind,
  XSDAUX.unsignedInt amount,
  UnitMultiplierType amountMultiplier optional
}

type DC_EVChargeParameterType DC_EVChargeParameter;

type record DC_EVChargeParameterType {
  XSDAUX.unsignedInt departureTime optional,
  DC_EVStatusType dC_EVStatus,
  PhysicalValueType eVMaximumCurrentLimit,
  PhysicalValueType eVMaximumPowerLimit optional,
  PhysicalValueType eVMaximumVoltageLimit,
  PhysicalValueType eVEnergyCapacity optional,
  PhysicalValueType eVEnergyRequest optional,
  PercentValueType fullSOC optional,
  PercentValueType bulkSOC optional
}

type enumerated DC_EVErrorCodeType {
  nO_ERROR,
  fAILED_RESSTemperatureInhibit,
  fAILED_EVShiftPosition,
  fAILED_ChargerConnectorLockFault,
  fAILED_EVRESSMalfunction,
  fAILED_ChargingCurrentdifferential,
  fAILED_ChargingVoltageOutOfRange,
  reserved_A,
  reserved_B,
  reserved_C,
  fAILED_ChargingSystemIncompatibility,
  noData
}

type DC_EVPowerDeliveryParameterType DC_EVPowerDeliveryParameter;
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
type record DC_EVPowerDeliveryParameterType {
  DC_EVStatusType dC_EVStatus,
  XSDAUX.Boolean bulkChargingComplete optional,
  XSDAUX.Boolean chargingComplete
}

type DC_EVSEChargeParameterType DC_EVSEChargeParameter;

type record DC_EVSEChargeParameterType {
  DC_EVSEStatusType dC_EVSEStatus,
  PhysicalValueType eVSEMaximumCurrentLimit,
  PhysicalValueType eVSEMaximumPowerLimit,
  PhysicalValueType eVSEMaximumVoltageLimit,
  PhysicalValueType eVSEMinimumCurrentLimit,
  PhysicalValueType eVSEMinimumVoltageLimit,
  PhysicalValueType eVSECurrentRegulationTolerance optional,
  PhysicalValueType eVSEPeakCurrentRipple,
  PhysicalValueType eVSEEnergyToBeDelivered optional
}

type DC_EVSEStatusType DC_EVSEStatus;

type enumerated DC_EVSEStatusCodeType {
  eVSE_NotReady,
  eVSE_Ready,
  eVSE_Shutdown,
  eVSE_UtilityInterruptEvent,
  eVSE_IsolationMonitoringActive,
  eVSE_EmergencyShutdown,
  eVSE_Malfunction,
  reserved_8,
  reserved_9,
  reserved_A,
  reserved_B,
  reserved_C
}

type record DC_EVSEStatusType {
  XSDAUX.unsignedShort notificationMaxDelay,
  EVSENotificationType eVSENotification,
  IsolationLevelType eVSEIsolationStatus optional,
  DC_EVSEStatusCodeType eVSEStatusCode
}

type DC_EVStatusType DC_EVStatus;

type record DC_EVStatusType {
  XSDAUX.Boolean eVReady,
  DC_EVErrorCodeType eVErrorCode,
  PercentValueType eVRESSSOC
}

type XSDAUX.base64Binary DHpublickeyType length (0 .. 88);

type record DiffieHellmanPublickeyType {
  XSDAUX.ID id,
  DHpublickeyType base
}

type record EMAIDType {
  XSDAUX.ID id,
  EMAIDType_1 base
}

type XSDAUX.string EMAIDType_1 length (14 .. 15);

type EVChargeParameterType_derivations EVChargeParameter;

type record EVChargeParameterType {
  XSDAUX.unsignedInt departureTime optional
}

type union EVChargeParameterType_derivations {
  @default
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
  EVChargeParameterType eVChargeParameterType,
  AC_EVChargeParameterType aC_EVChargeParameterType,
  DC_EVChargeParameterType dC_EVChargeParameterType
}

type union EVChargeParameter_group {
  @default
  EVChargeParameter eVChargeParameter,
  AC_EVChargeParameter aC_EVChargeParameter,
  DC_EVChargeParameter dC_EVChargeParameter
}

type EVPowerDeliveryParameterType_derivations EVPowerDeliveryParameter;

type record EVPowerDeliveryParameterType {
}

type union EVPowerDeliveryParameterType_derivations {
  @default
  EVPowerDeliveryParameterType eVPowerDeliveryParameterType,
  DC_EVPowerDeliveryParameterType dC_EVPowerDeliveryParameterType
}

type union EVPowerDeliveryParameter_group {
  @default
  EVPowerDeliveryParameter eVPowerDeliveryParameter,
  DC_EVPowerDeliveryParameter dC_EVPowerDeliveryParameter
}

type EVSEChargeParameterType_derivations EVSEChargeParameter;

type record EVSEChargeParameterType {
}

type union EVSEChargeParameterType_derivations {
  @default
  EVSEChargeParameterType eVSEChargeParameterType,
  AC_EVSEChargeParameterType aC_EVSEChargeParameterType,
  DC_EVSEChargeParameterType dC_EVSEChargeParameterType
}

type union EVSEChargeParameter_group {
  @default
  EVSEChargeParameter eVSEChargeParameter,
  AC_EVSEChargeParameter aC_EVSEChargeParameter,
  DC_EVSEChargeParameter dC_EVSEChargeParameter
}

type enumerated EVSENotificationType {
  none_,
  stopCharging,
  reNegotiation
}

type enumerated EVSEProcessingType {
  finished,
  ongoing,
  ongoing_WaitingForCustomerInteraction
}

type EVSEStatusType_derivations EVSEStatus;

type record EVSEStatusType {
  XSDAUX.unsignedShort notificationMaxDelay,
  EVSENotificationType eVSENotification
}

type union EVSEStatusType_derivations {
  @default
  EVSEStatusType eVSEStatusType,
  AC_EVSEStatusType aC_EVSEStatusType,
  DC_EVSEStatusType dC_EVSEStatusType
}

type union EVSEStatus_group {
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  @default
  EVSEStatus eVSEStatus,
  AC_EVSEStatus aC_EVSEStatus,
  DC_EVSEStatus dC_EVSEStatus
}

type EVStatusType_derivations EVStatus;

type record EVStatusType {
}

type union EVStatusType_derivations {
  @default
  EVStatusType eVStatusType,
  DC_EVStatusType dC_EVStatusType
}

type union EVStatus_group {
  @default
  EVStatus eVStatus,
  DC_EVStatus dC_EVStatus
}

type enumerated EnergyTransferModeType {
  aC_single_phase_core,
  aC_three_phase_core,
  dC_core,
  dC_extended,
  dC_combo_core,
  dC_unique
}

type EntryType_derivations Entry;

type record EntryType {
  TimeInterval_group timeInterval
}

type union EntryType_derivations {
  @default
  EntryType entryType,
  PMaxScheduleEntryType pMaxScheduleEntryType,
  SalesTariffEntryType salesTariffEntryType
}

type union Entry_group {
  @default
  Entry entry,
  PMaxScheduleEntry pMaxScheduleEntry,
  SalesTariffEntry salesTariffEntry
}

type XSDAUX.hexBinary EvccIDType length (0 .. 6);

type XSDAUX.string EvseIDType length (7 .. 37);

type enumerated FaultCodeType {
  parsingError,
  noTLSRootCertificatAvailable,
  unknownError
}

type XSDAUX.string FaultMsgType length (0 .. 64);

type XSDAUX.base64Binary GenChallengeType length (24);

type record IntervalType {
}

type union IntervalType_derivations {
  @default
  IntervalType intervalType,
  RelativeTimeIntervalType relativeTimeIntervalType
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  type enumerated IsolationLevelType {
    invalid,
    valid,
    warning,
    fault,
    no_IMD
  }

  type record ListOfRootCertificateIDsType {
    record    length    (1    ..    20)    of    http_www_w3_org_2000_09_xmldsig.X509IssuerSerialType
rootCertificateID_list
  }

  type XSDAUX.byte MaxNumPhasesType (1 .. 3);

  type XSDAUX.string MeterIDType length (0 .. 32);

  type record MeterInfoType {
    MeterIDType meterID,
    XSDAUX.unsignedLong meterReading optional,
    SigMeterReadingType sigMeterReading optional,
    MeterStatusType meterStatus optional,
    XSDAUX.long tMeter optional
  }

  type XSDAUX.short MeterStatusType;

  type record NotificationType {
    FaultCodeType faultCode,
    FaultMsgType faultMsg optional
  }

  type PMaxScheduleEntryType PMaxScheduleEntry;

  type record PMaxScheduleEntryType {
    TimeInterval_group timeInterval,
    PhysicalValueType pMax
  }

  type record PMaxScheduleType {
    record length (1 .. 1024) of PMaxScheduleEntryType pMaxScheduleEntry_list
  }

  type record ParameterSetType {
    XSDAUX.short parameterSetID,
    record length (1 .. 16) of ParameterType parameter_list
  }

  type record ParameterType {
    XSDAUX.string name,
    union {
      XSDAUX.Boolean boolValue,
      XSDAUX.byte byteValue,
      XSDAUX.short shortValue,
      XSDAUX.int intValue,
      PhysicalValueType physicalValue,
      XSDAUX.string stringValue
    } choice
  }

  type record PaymentOptionListType {
    record length (1 .. 2) of PaymentOptionType paymentOption_list
  }

  type enumerated PaymentOptionType {
    contract,
    externalPayment
  }

  type XSDAUX.byte PercentValueType (0 .. 100);

  type record PhysicalValueType {
    UnitMultiplierType multiplier,
    UnitSymbolType unit,
    XSDAUX.short value_
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}

type XSDAUX.base64Binary PrivateKeyType length (0 .. 64);

type record ProfileEntryType {
  XSDAUX.unsignedInt chargingProfileEntryStart,
  PhysicalValueType chargingProfileEntryMaxPower,
  MaxNumPhasesType chargingProfileEntryMaxNumberOfPhasesInUse optional
}

type RelativeTimeIntervalType RelativeTimeInterval;

type record RelativeTimeIntervalType {
  XSDAUX.unsignedInt start_ (0 .. 16777214),
  XSDAUX.unsignedInt duration (0 .. 86400) optional
}

type enumerated ResponseCodeType {
  oK,
  oK_NewSessionEstablished,
  oK_OldSessionJoined,
  oK_CertificateExpiresSoon,
  fAILED,
  fAILED_SequenceError,
  fAILED_ServiceIDInvalid,
  fAILED_UnknownSession,
  fAILED_ServiceSelectionInvalid,
  fAILED_PaymentSelectionInvalid,
  fAILED_CertificateExpired,
  fAILED_SignatureError,
  fAILED_NoCertificateAvailable,
  fAILED_CertChainError,
  fAILED_ChallengeInvalid,
  fAILED_ContractCanceled,
  fAILED_WrongChargeParameter,
  fAILED_PowerDeliveryNotApplied,
  fAILED_TariffSelectionInvalid,
  fAILED_ChargingProfileInvalid,
  fAILED_MeteringSignatureNotValid,
  fAILED_NoChargeServiceSelected,
  fAILED_WrongEnergyTransferMode,
  fAILED_ContactorError,
  fAILED_CertificateNotAllowedAtThisEVSE,
  fAILED_CertificateRevoked
}

type XSDAUX.unsignedByte SAIDType (1 .. 255);

type SAScheduleListType SAScheduleList;

type record SAScheduleListType {
  record length (1 .. 3) of SAScheduleTupleType sAScheduleTuple_list
}

type record SAScheduleTupleType {
  SAIDType sAScheduleTupleID,
  PMaxScheduleType pMaxSchedule,
  SalesTariffType salesTariff optional
}

type SASchedulesType_derivations SASchedules;

type record SASchedulesType {
}

type union SASchedulesType_derivations {
  @default
  SASchedulesType sASchedulesType,
  SAScheduleListType sAScheduleListType
}

type union SASchedules_group {
  @default
  SASchedules sASchedules,
  SAScheduleList sAScheduleList
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
}

type SalesTariffEntryType SalesTariffEntry;

type record SalesTariffEntryType {
  TimeInterval_group timeInterval,
  XSDAUX.unsignedByte ePriceLevel optional,
  record length (0 .. 3) of ConsumptionCostType consumptionCost_list
}

type record SalesTariffType {
  XSDAUX.ID id optional,
  SAIDType salesTariffID,
  TariffDescriptionType salesTariffDescription optional,
  XSDAUX.unsignedByte numEPriceLevels optional,
  record length (1 .. 1024) of SalesTariffEntryType salesTariffEntry_list
}

type record SelectedServiceListType {
  record length (1 .. 16) of SelectedServiceType selectedService_list
}

type record SelectedServiceType {
  ServiceIDType serviceID,
  XSDAUX.short parameterSetID optional
}

type enumerated ServiceCategoryType {
  eVCharging,
  internet,
  contractCertificate,
  otherCustom
}

type XSDAUX.unsignedShort ServiceIDType;

type record ServiceListType {
  record length (1 .. 8) of ServiceType_derivations service_list
}

type XSDAUX.string ServiceNameType length (0 .. 32);

type record ServiceParameterListType {
  record length (1 .. 255) of ParameterSetType parameterSet_list
}

type XSDAUX.string ServiceScopeType length (0 .. 64);

type record ServiceType {
  ServiceIDType serviceID,
  ServiceNameType serviceName optional,
  ServiceCategoryType serviceCategory,
  ServiceScopeType serviceScope optional,
  XSDAUX.Boolean freeService
}

type union ServiceType_derivations {
  @default
  ServiceType serviceType,
  ChargeServiceType chargeServiceType
}

type XSDAUX.hexBinary SessionIDType length (0 .. 8);

type XSDAUX.base64Binary SigMeterReadingType length (0 .. 64);

type record SubCertificatesType {
  record length (1 .. 4) of urn_iso_15118_2_2013_X509Def.X509CertificateType x509Certificate_list
}

type record SupportedEnergyTransferModeType {
  record length (1 .. 6) of EnergyTransferModeType energyTransferMode_list
}

type XSDAUX.string TariffDescriptionType length (0 .. 32);
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  type IntervalType_derivations TimeInterval;

  type union TimeInterval_group {
    @default
    TimeInterval timeInterval,
    RelativeTimeInterval relativeTimeInterval
  }

  type XSDAUX.byte UnitMultiplierType (-3 .. 3);

  type enumerated UnitSymbolType {
    h,
    m,
    s,
    a,
    v,
    w,
    wh
  }

  type enumerated ValueType {
    bool,
    byte,
    short,
    int,
    physicalValue,
    string
  }

}
with {
  encode "V2G_CI_MsgDef_ISO";
}
```

```
module urn_iso_15118_2_2013_MsgDef {
  import from urn_iso_15118_2_2013_MsgHeader language "XSD" all;

  import from urn_iso_15118_2_2013_MsgBody language "XSD" all;

  type record V2G_Message {
    urn_iso_15118_2_2013_MsgHeader.MessageHeaderType header,
    urn_iso_15118_2_2013_MsgBody.BodyType body
  }

}
with {
  encode "V2G_CI_MsgDef_ISO";
}
```

```
module urn_iso_15118_2_2010_AppProtocol {
  import from XSDAUX all;

  type record AppProtocolType {
    ProtocolNamespaceType protocolNamespace,
    XSDAUX.unsignedInt versionNumberMajor,
    XSDAUX.unsignedInt versionNumberMinor,
    IdType schemaID,
    PriorityType priority
  }

  type XSDAUX.unsignedByte IdType;

  type XSDAUX.unsignedByte PriorityType (1 .. 20);

  type XSDAUX.string ProtocolNameType length (0 .. 30);

  type XSDAUX.anyURI ProtocolNamespaceType length (0 .. 100);
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  type enumerated ResponseCodeType {
    oK_SuccessfulNegotiation,
    oK_SuccessfulNegotiationWithMinorDeviation,
    failed_NoNegotiation
  }

  type record SupportedAppProtocolReq {
    record length (1 .. 20) of AppProtocolType appProtocol_list
  }

  type record SupportedAppProtocolRes {
    ResponseCodeType responseCode,
    IdType schemaID optional
  }

}
with {
  encode "V2G_CI_AppProtocol_ISO";
}




module urn_iso_15118_2_2013_MsgBody {
  import from urn_iso_15118_2_2013_X509Def language "XSD" all;

  import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;

  import from XSDAUX all;

  type AuthorizationReqType AuthorizationReq;

  type record AuthorizationReqType {
    XSDAUX.ID id optional,
    urn_iso_15118_2_2013_MsgDataTypes.GenChallengeType genChallenge optional
  }

  type AuthorizationResType AuthorizationRes;

  type record AuthorizationResType {
    urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
    urn_iso_15118_2_2013_MsgDataTypes.EVSEProcessingType eVSEProcessing
  }

  type record BodyBaseType {
  }

  type union BodyBaseType_derivations {
    @default
    BodyBaseType bodyBaseType,
    AuthorizationReqType authorizationReqType,
    AuthorizationResType authorizationResType,
    CableCheckReqType cableCheckReqType,
    CableCheckResType cableCheckResType,
    CertificateInstallationReqType certificateInstallationReqType,
    CertificateInstallationResType certificateInstallationResType,
    CertificateUpdateReqType certificateUpdateReqType,
    CertificateUpdateResType certificateUpdateResType,
    ChargeParameterDiscoveryReqType chargeParameterDiscoveryReqType,
    ChargeParameterDiscoveryResType chargeParameterDiscoveryResType,
    ChargingStatusReqType chargingStatusReqType,
    ChargingStatusResType chargingStatusResType,
    CurrentDemandReqType currentDemandReqType,
    CurrentDemandResType currentDemandResType,
    MeteringReceiptReqType meteringReceiptReqType,
    MeteringReceiptResType meteringReceiptResType,
    PaymentDetailsReqType paymentDetailsReqType,
    PaymentDetailsResType paymentDetailsResType,
    PaymentServiceSelectionReqType paymentServiceSelectionReqType,
    PaymentServiceSelectionResType paymentServiceSelectionResType,
    PowerDeliveryReqType powerDeliveryReqType,
    PowerDeliveryResType powerDeliveryResType,
    PreChargeReqType preChargeReqType,
    PreChargeResType preChargeResType,
    ServiceDetailReqType serviceDetailReqType,
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  ServiceDetailResType serviceDetailResType,
  ServiceDiscoveryReqType serviceDiscoveryReqType,
  ServiceDiscoveryResType serviceDiscoveryResType,
  SessionSetupReqType sessionSetupReqType,
  SessionSetupResType sessionSetupResType,
  SessionStopReqType sessionStopReqType,
  SessionStopResType sessionStopResType,
  WeldingDetectionReqType weldingDetectionReqType,
  WeldingDetectionResType weldingDetectionResType
}

type BodyBaseType_derivations BodyElement;

type union BodyElement_group {
  @default
  BodyElement bodyElement,
  AuthorizationReq authorizationReq,
  AuthorizationRes authorizationRes,
  CableCheckReq cableCheckReq,
  CableCheckRes cableCheckRes,
  CertificateInstallationReq certificateInstallationReq,
  CertificateInstallationRes certificateInstallationRes,
  CertificateUpdateReq certificateUpdateReq,
  CertificateUpdateRes certificateUpdateRes,
  ChargeParameterDiscoveryReq chargeParameterDiscoveryReq,
  ChargeParameterDiscoveryRes chargeParameterDiscoveryRes,
  ChargingStatusReq chargingStatusReq,
  ChargingStatusRes chargingStatusRes,
  CurrentDemandReq currentDemandReq,
  CurrentDemandRes currentDemandRes,
  MeteringReceiptReq meteringReceiptReq,
  MeteringReceiptRes meteringReceiptRes,
  PaymentDetailsReq paymentDetailsReq,
  PaymentDetailsRes paymentDetailsRes,
  PaymentServiceSelectionReq paymentServiceSelectionReq,
  PaymentServiceSelectionRes paymentServiceSelectionRes,
  PowerDeliveryReq powerDeliveryReq,
  PowerDeliveryRes powerDeliveryRes,
  PreChargeReq preChargeReq,
  PreChargeRes preChargeRes,
  ServiceDetailReq serviceDetailReq,
  ServiceDetailRes serviceDetailRes,
  ServiceDiscoveryReq serviceDiscoveryReq,
  ServiceDiscoveryRes serviceDiscoveryRes,
  SessionSetupReq sessionSetupReq,
  SessionSetupRes sessionSetupRes,
  SessionStopReq sessionStopReq,
  SessionStopRes sessionStopRes,
  WeldingDetectionReq weldingDetectionReq,
  WeldingDetectionRes weldingDetectionRes
}

type record BodyType {
  BodyElement_group bodyElement optional
}

type CableCheckReqType CableCheckReq;

type record CableCheckReqType {
  urn_iso_15118_2_2013_MsgDataTypes.DC_EVStatusType dC_EVStatus
}

type CableCheckResType CableCheckRes;

type record CableCheckResType {
  urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
  urn_iso_15118_2_2013_MsgDataTypes.DC_EVSEStatusType dC_EVSEStatus,
  urn_iso_15118_2_2013_MsgDataTypes.EVSEProcessingType eVSEProcessing
}

type CertificateInstallationReqType CertificateInstallationReq;

type record CertificateInstallationReqType {
  XSDAUX.ID id,
  urn_iso_15118_2_2013_X509Def.X509CertificateType oEMProvisioningCert,
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
    urn_iso_15118_2_2013_MsgDataTypes.ListOfRootCertificateIDsType listOfRootCertificateIDs
  }

  type CertificateInstallationResType CertificateInstallationRes;

  type record CertificateInstallationResType {
    urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
    urn_iso_15118_2_2013_MsgDataTypes.CertificateChainType sAProvisioningCertificateChain,
    urn_iso_15118_2_2013_MsgDataTypes.CertificateChainType contractSignatureCertChain,
    urn_iso_15118_2_2013_MsgDataTypes.ContractSignatureEncryptedPrivateKeyType
contractSignatureEncryptedPrivateKey,
    urn_iso_15118_2_2013_MsgDataTypes.DiffieHellmanPublickeyType dHpublickey,
    urn_iso_15118_2_2013_MsgDataTypes.EMAIDType eMAID
  }

  type CertificateUpdateReqType CertificateUpdateReq;

  type record CertificateUpdateReqType {
    XSDAUX.ID id,
    urn_iso_15118_2_2013_MsgDataTypes.CertificateChainType contractSignatureCertChain,
    urn_iso_15118_2_2013_MsgDataTypes.EMAIDType_1 eMAID,
    urn_iso_15118_2_2013_MsgDataTypes.ListOfRootCertificateIDsType listOfRootCertificateIDs
  }

  type CertificateUpdateResType CertificateUpdateRes;

  type record CertificateUpdateResType {
    urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
    urn_iso_15118_2_2013_MsgDataTypes.CertificateChainType sAProvisioningCertificateChain,
    urn_iso_15118_2_2013_MsgDataTypes.CertificateChainType contractSignatureCertChain,
    urn_iso_15118_2_2013_MsgDataTypes.ContractSignatureEncryptedPrivateKeyType
contractSignatureEncryptedPrivateKey,
    urn_iso_15118_2_2013_MsgDataTypes.DiffieHellmanPublickeyType dHpublickey,
    urn_iso_15118_2_2013_MsgDataTypes.EMAIDType eMAID,
    XSDAUX.short retryCounter optional
  }

  type ChargeParameterDiscoveryReqType ChargeParameterDiscoveryReq;

  type record ChargeParameterDiscoveryReqType {
    XSDAUX.unsignedShort maxEntriesSAScheduleTuple optional,
    urn_iso_15118_2_2013_MsgDataTypes.EnergyTransferModeType requestedEnergyTransferMode,
    urn_iso_15118_2_2013_MsgDataTypes.EVChargeParameter_group eVChargeParameter
  }

  type ChargeParameterDiscoveryResType ChargeParameterDiscoveryRes;

  type record ChargeParameterDiscoveryResType {
    urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
    urn_iso_15118_2_2013_MsgDataTypes.EVSEProcessingType eVSEProcessing,
    urn_iso_15118_2_2013_MsgDataTypes.SASchedules_group sASchedules optional,
    urn_iso_15118_2_2013_MsgDataTypes.EVSEChargeParameter_group eVSEChargeParameter
  }

  type ChargingStatusReqType ChargingStatusReq;

  type record ChargingStatusReqType {
  }

  type ChargingStatusResType ChargingStatusRes;

  type record ChargingStatusResType {
    urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
    urn_iso_15118_2_2013_MsgDataTypes.EvseIDType eVSEID,
    urn_iso_15118_2_2013_MsgDataTypes.SAIDType sAScheduleTupleID,
    urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVSEMaxCurrent optional,
    urn_iso_15118_2_2013_MsgDataTypes.MeterInfoType meterInfo optional,
    XSDAUX.Boolean receiptRequired optional,
    urn_iso_15118_2_2013_MsgDataTypes.AC_EVSEStatusType aC_EVSEStatus
  }

  type CurrentDemandReqType CurrentDemandReq;

  type record CurrentDemandReqType {
    urn_iso_15118_2_2013_MsgDataTypes.DC_EVStatusType dC_EVStatus,
```


© ISO 2024 – All rights reserved
Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVTargetCurrent,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVMaximumVoltageLimit optional,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVMaximumCurrentLimit optional,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVMaximumPowerLimit optional,
  XSDAUX.Boolean bulkChargingComplete optional,
  XSDAUX.Boolean chargingComplete,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType remainingTimeToFullSoC optional,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType remainingTimeToBulkSoC optional,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVTargetVoltage
}

type CurrentDemandResType CurrentDemandRes;

type record CurrentDemandResType {
  urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
  urn_iso_15118_2_2013_MsgDataTypes.DC_EVSEStatusType dC_EVSEStatus,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVSEPresentVoltage,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVSEPresentCurrent,
  XSDAUX.Boolean eVSECurrentLimitAchieved,
  XSDAUX.Boolean eVSEVoltageLimitAchieved,
  XSDAUX.Boolean eVSEPowerLimitAchieved,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVSEMaximumVoltageLimit optional,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVSEMaximumCurrentLimit optional,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVSEMaximumPowerLimit optional,
  urn_iso_15118_2_2013_MsgDataTypes.EvseIDType eVSEID,
  urn_iso_15118_2_2013_MsgDataTypes.SAIDType sAScheduleTupleID,
  urn_iso_15118_2_2013_MsgDataTypes.MeterInfoType meterInfo optional,
  XSDAUX.Boolean receiptRequired optional
}

type MeteringReceiptReqType MeteringReceiptReq;

type record MeteringReceiptReqType {
  XSDAUX.ID id optional,
  urn_iso_15118_2_2013_MsgDataTypes.SessionIDType sessionID,
  urn_iso_15118_2_2013_MsgDataTypes.SAIDType sAScheduleTupleID optional,
  urn_iso_15118_2_2013_MsgDataTypes.MeterInfoType meterInfo
}

type MeteringReceiptResType MeteringReceiptRes;

type record MeteringReceiptResType {
  urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
  urn_iso_15118_2_2013_MsgDataTypes.EVSEStatus_group eVSEStatus
}

type PaymentDetailsReqType PaymentDetailsReq;

type record PaymentDetailsReqType {
  urn_iso_15118_2_2013_MsgDataTypes.EMAIDType_1 eMAID,
  urn_iso_15118_2_2013_MsgDataTypes.CertificateChainType contractSignatureCertChain
}

type PaymentDetailsResType PaymentDetailsRes;

type record PaymentDetailsResType {
  urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
  urn_iso_15118_2_2013_MsgDataTypes.GenChallengeType genChallenge,
  XSDAUX.long eVSETimeStamp
}

type PaymentServiceSelectionReqType PaymentServiceSelectionReq;

type record PaymentServiceSelectionReqType {
  urn_iso_15118_2_2013_MsgDataTypes.PaymentOptionType selectedPaymentOption,
  urn_iso_15118_2_2013_MsgDataTypes.SelectedServiceListType selectedServiceList
}

type PaymentServiceSelectionResType PaymentServiceSelectionRes;

type record PaymentServiceSelectionResType {
  urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode
}

type PowerDeliveryReqType PowerDeliveryReq;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
type record PowerDeliveryReqType {
  urn_iso_15118_2_2013_MsgDataTypes.ChargeProgressType chargeProgress,
  urn_iso_15118_2_2013_MsgDataTypes.SAIDType sAScheduleTupleID,
  urn_iso_15118_2_2013_MsgDataTypes.ChargingProfileType chargingProfile optional,
  urn_iso_15118_2_2013_MsgDataTypes.EVPowerDeliveryParameter_group eVPowerDeliveryParameter optional
}

type PowerDeliveryResType PowerDeliveryRes;

type record PowerDeliveryResType {
  urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
  urn_iso_15118_2_2013_MsgDataTypes.EVSEStatus_group eVSEStatus
}

type PreChargeReqType PreChargeReq;

type record PreChargeReqType {
  urn_iso_15118_2_2013_MsgDataTypes.DC_EVStatusType dC_EVStatus,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVTargetVoltage,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVTargetCurrent
}

type PreChargeResType PreChargeRes;

type record PreChargeResType {
  urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
  urn_iso_15118_2_2013_MsgDataTypes.DC_EVSEStatusType dC_EVSEStatus,
  urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVSEPresentVoltage
}

type ServiceDetailReqType ServiceDetailReq;

type record ServiceDetailReqType {
  urn_iso_15118_2_2013_MsgDataTypes.ServiceIDType serviceID
}

type ServiceDetailResType ServiceDetailRes;

type record ServiceDetailResType {
  urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
  urn_iso_15118_2_2013_MsgDataTypes.ServiceIDType serviceID,
  urn_iso_15118_2_2013_MsgDataTypes.ServiceParameterListType serviceParameterList optional
}

type ServiceDiscoveryReqType ServiceDiscoveryReq;

type record ServiceDiscoveryReqType {
  urn_iso_15118_2_2013_MsgDataTypes.ServiceScopeType serviceScope optional,
  urn_iso_15118_2_2013_MsgDataTypes.ServiceCategoryType serviceCategory optional
}

type ServiceDiscoveryResType ServiceDiscoveryRes;

type record ServiceDiscoveryResType {
  urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
  urn_iso_15118_2_2013_MsgDataTypes.PaymentOptionListType paymentOptionList,
  urn_iso_15118_2_2013_MsgDataTypes.ChargeServiceType chargeService,
  urn_iso_15118_2_2013_MsgDataTypes.ServiceListType serviceList optional
}

type SessionSetupReqType SessionSetupReq;

type record SessionSetupReqType {
  urn_iso_15118_2_2013_MsgDataTypes.EvccIDType eVCCID
}

type SessionSetupResType SessionSetupRes;

type record SessionSetupResType {
  urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
  urn_iso_15118_2_2013_MsgDataTypes.EvseIDType eVSEID,
  XSDAUX.long eVSETimeStamp optional
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled


```
  type SessionStopReqType SessionStopReq;

  type record SessionStopReqType {
    urn_iso_15118_2_2013_MsgDataTypes.ChargingSessionType chargingSession
  }

  type SessionStopResType SessionStopRes;

  type record SessionStopResType {
    urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode
  }

  type WeldingDetectionReqType WeldingDetectionReq;

  type record WeldingDetectionReqType {
    urn_iso_15118_2_2013_MsgDataTypes.DC_EVStatusType dC_EVStatus
  }

  type WeldingDetectionResType WeldingDetectionRes;

  type record WeldingDetectionResType {
    urn_iso_15118_2_2013_MsgDataTypes.ResponseCodeType responseCode,
    urn_iso_15118_2_2013_MsgDataTypes.DC_EVSEStatusType dC_EVSEStatus,
    urn_iso_15118_2_2013_MsgDataTypes.PhysicalValueType eVSEPresentVoltage
  }

}
with {
  encode "V2G_CI_MsgDef_ISO";
}



module http_www_w3_org_2000_09_xmldsig {
  import from XSDAUX all;

  type CanonicalizationMethodType CanonicalizationMethod;

  type record CanonicalizationMethodType {
    record of XSDAUX.string embed_values optional,
    XSDAUX.anyURI algorithm,
    record of XSDAUX.AnyType elem_list
  }

  type XSDAUX.base64Binary CryptoBinary;

  type DSAKeyValueType DSAKeyValue;

  type record DSAKeyValueType {
    record {
      CryptoBinary p,
      CryptoBinary q
    } sequence optional,
    CryptoBinary g optional,
    CryptoBinary y,
    CryptoBinary j optional,
    record {
      CryptoBinary seed,
      CryptoBinary pgenCounter
    } sequence_1 optional
  }

  type DigestMethodType DigestMethod;

  type record DigestMethodType {
    record of XSDAUX.string embed_values optional,
    XSDAUX.anyURI algorithm,
    record of XSDAUX.AnyType elem_list
  }

  type DigestValueType DigestValue;

  type XSDAUX.base64Binary DigestValueType;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
type XSDAUX.integerXSD HMACOutputLengthType;

type KeyInfoType KeyInfo;

type record KeyInfoType {
  record of XSDAUX.string embed_values optional,
  XSDAUX.ID id optional,
  record length (1 .. infinity) of union {
    XSDAUX.string keyName,
    KeyValueType keyValue,
    RetrievalMethodType retrievalMethod,
    X509DataType x509Data,
    PGPDataType pGPData,
    SPKIDataType sPKIData,
    XSDAUX.string mgmtData,
    XSDAUX.AnyType elem
  } choice_list
}

type XSDAUX.string KeyName;

type KeyValueType KeyValue;

type record KeyValueType {
  record of XSDAUX.string embed_values optional,
  union {
    DSAKeyValueType dSAKeyValue,
    RSAKeyValueType rSAKeyValue,
    XSDAUX.AnyType elem
  } choice
}

type ManifestType Manifest;

type record ManifestType {
  XSDAUX.ID id optional,
  record length (1 .. infinity) of ReferenceType reference_list
}

type XSDAUX.string MgmtData;

type ObjectType Object;

type record ObjectType {
  record of XSDAUX.string embed_values optional,
  XSDAUX.anyURI encoding optional,
  XSDAUX.ID id optional,
  XSDAUX.string mimeType optional,
  record of record {
    XSDAUX.AnyType elem
  } sequence_list
}

type PGPDataType PGPData;

type record PGPDataType {
  union {
    record {
      XSDAUX.base64Binary pGPKeyID,
      XSDAUX.base64Binary pGPKeyPacket optional,
      record of XSDAUX.AnyType elem_list
    } sequence,
    record {
      XSDAUX.base64Binary pGPKeyPacket,
      record of XSDAUX.AnyType elem_list
    } sequence_1
  } choice
}

type RSAKeyValueType RSAKeyValue;

type record RSAKeyValueType {
  CryptoBinary modulus,
  CryptoBinary exponent
}
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
type ReferenceType Reference;

type record ReferenceType {
  XSDAUX.ID id optional,
  XSDAUX.anyURI type_ optional,
  XSDAUX.anyURI uRI optional,
  TransformsType transforms optional,
  DigestMethodType digestMethod,
  DigestValueType digestValue
}

type RetrievalMethodType RetrievalMethod;

type record RetrievalMethodType {
  XSDAUX.anyURI type_ optional,
  XSDAUX.anyURI uRI optional,
  TransformsType transforms optional
}

type SPKIDataType SPKIData;

type record SPKIDataType {
  record length (1 .. infinity) of record {
    XSDAUX.base64Binary sPKISexp,
    XSDAUX.AnyType elem optional
  } sequence_list
}

type SignatureType Signature;

type SignatureMethodType SignatureMethod;

type record SignatureMethodType {
  record of XSDAUX.string embed_values optional,
  XSDAUX.anyURI algorithm,
  HMACOutputLengthType hMACOutputLength optional,
  record of XSDAUX.AnyType elem_list
}

type SignaturePropertiesType SignatureProperties;

type record SignaturePropertiesType {
  XSDAUX.ID id optional,
  record length (1 .. infinity) of SignaturePropertyType signatureProperty_list
}

type SignaturePropertyType SignatureProperty;

type record SignaturePropertyType {
  record of XSDAUX.string embed_values optional,
  XSDAUX.ID id optional,
  XSDAUX.anyURI target,
  record length (1 .. infinity) of union {
    XSDAUX.AnyType elem
  } choice_list
}

type record SignatureType {
  XSDAUX.ID id optional,
  SignedInfoType signedInfo,
  SignatureValueType signatureValue,
  KeyInfoType keyInfo optional,
  record of ObjectType object_list optional
}

type SignatureValueType SignatureValue;

type record SignatureValueType {
  XSDAUX.ID id optional,
  XSDAUX.base64Binary base
}

type SignedInfoType SignedInfo;
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
  type record SignedInfoType {
    XSDAUX.ID id optional,
    CanonicalizationMethodType canonicalizationMethod,
    SignatureMethodType signatureMethod,
    record length (1 .. infinity) of ReferenceType reference_list
  }

  type TransformType Transform;

  type record TransformType {
    record of XSDAUX.string embed_values optional,
    XSDAUX.anyURI algorithm,
    record of union {
      XSDAUX.AnyType elem,
      XSDAUX.string xPath
    } choice_list
  }

  type TransformsType Transforms;

  type record TransformsType {
    record length (1 .. infinity) of TransformType transform_list
  }

  type X509DataType X509Data;

  type record X509DataType {
    record length (1 .. infinity) of record {
      union {
        X509IssuerSerialType x509IssuerSerial,
        XSDAUX.base64Binary x509SKI,
        XSDAUX.string x509SubjectName,
        XSDAUX.base64Binary x509Certificate,
        XSDAUX.base64Binary x509CRL,
        XSDAUX.AnyType elem
      } choice
    } sequence_list
  }

  type record X509IssuerSerialType {
    XSDAUX.string x509IssuerName,
    XSDAUX.integerXSD x509SerialNumber
  }

}
with {
  encode "V2G_CI_MsgDef_ISO";
}



module DataStructure_V2G {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;

    type integer StartType (0 .. 16777214);
    type integer DurationType (0 .. 86400);
    type record of StartType StartList;
    type record of PhysicalValueType PmaxList;
}
```

P.8 Data types for Security

```
module DataStructure_Security {

    import from urn_iso_15118_2_2013_MsgDataTypes language "XSD" all;
    import from http_www_w3_org_2000_09_xmldsig language "XSD" {
        type ReferenceType;
    }
    import from urn_iso_15118_2_2013_X509Def all;
```


© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled


```
    type record IdDigestElement {charstring id, charstring digestValue};
    type set of IdDigestElement idDigestValueMap;
    type record of charstring CertificatesList;
    type record length (0 .. infinity) of ReferenceType LocalReferenceList;
    type record IdCertificateElement {charstring id, X509CertificateType certificate};
    type set of IdCertificateElement idCertificateMap;
    type record IdPrivateKeyElement {charstring id, charstring privateKey};
    type set of IdPrivateKeyElement idPrivateKeyMap;
}
```

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47
Printed copies are uncontrolled

Annex Q
(informative)

Overview of changed test cases compared to ISO 15118-4:2018

Table 856 provides an overview of all new test cases in this document compared to ISO 15118-4:2018.

Table 856 — List of new test cases in this document

EVCC Test Cases	SECC Test Cases
TC_EVCC_AC_VTB_Reliability_001	TC_SECC_AC_VTB_Reliability_001
TC_EVCC_AC_VTB_Reliability_002	TC_SECC_AC_VTB_Reliability_002
TC_EVCC_AC_VTB_Reliability_003	TC_SECC_AC_VTB_Reliability_003
TC_EVCC_AC_VTB_Reliability_005	TC_SECC_DC_VTB_Reliability_001
TC_EVCC_AC_VTB_Reliability_006	TC_SECC_DC_VTB_Reliability_002
TC_EVCC_AC_VTB_Reliability_007	TC_SECC_DC_VTB_Reliability_003
TC_EVCC_DC_VTB_Reliability_001	
TC_EVCC_DC_VTB_Reliability_002	
TC_EVCC_DC_VTB_Reliability_003	
TC_EVCC_DC_VTB_Reliability_005	
TC_EVCC_DC_VTB_Reliability_006	
TC_EVCC_DC_VTB_Reliability_007	

Table 857 provides an overview of all test cases that were deleted in this document compared to ISO 15118-4:2018.

Table 857 — List of deleted test cases in this document

EVCC Test Cases	SECC Test Cases
	TC_SECC_CMN_VTB_SupportedApPProtocol_003
	TC_SECC_CMN_VTB_SessionSetup_003
	TC_SECC_CMN_VTB_CertificateUpdate_007

Table 858 provides an overview of all updated test cases in this document compared to ISO 15118-4:2018.

Table 858 — List of updated test cases in this document

EVCC Test Cases	SECC Test Cases
TC_EVCC_AC_VTB_SDP_001	TC_SECC_CMN_VTB_SupportedAppProtocol_002
TC_EVCC_AC_VTB_SDP_002	TC_SECC_CMN_VTB_SessionSetup_002
TC_EVCC_AC_VTB_SDP_003	TC_SECC_AC_VTB_SessionSetup_001

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

TC_EVCC_AC_VTB_SDP_004	TC_SECC_AC_VTB_SessionSetup_002
TC_EVCC_AC_VTB_SDP_005	TC_SECC_AC_VTB_SessionSetup_003
TC_EVCC_AC_VTB_SDP_006	TC_SECC_AC_VTB_SessionSetup_004
TC_EVCC_AC_VTB_SDP_007	TC_SECC_AC_VTB_SessionSetup_005
TC_EVCC_AC_VTB_SDP_008	TC_SECC_AC_VTB_SessionSetup_006
TC_EVCC_AC_VTB_SDP_009	TC_SECC_DC_VTB_SessionSetup_001
TC_EVCC_AC_VTB_SDP_010	TC_SECC_DC_VTB_SessionSetup_002
TC_EVCC_AC_VTB_SDP_011	TC_SECC_DC_VTB_SessionSetup_003
TC_EVCC_AC_VTB_SDP_012	TC_SECC_DC_VTB_SessionSetup_004
TC_EVCC_AC_VTB_SDP_013	TC_SECC_DC_VTB_SessionSetup_005
TC_EVCC_AC_VTB_SDP_014	TC_SECC_DC_VTB_SessionSetup_006
TC_EVCC_AC_VTB_SDP_015	TC_SECC_CMN_VTB_V2GTPSessionSetup_002
TC_EVCC_AC_VTB_SDP_016	TC_SECC_CMN_VTB_V2GTPSessionSetup_005
TC_EVCC_DC_VTB_SDP_001	TC_SECC_CMN_VTB_ServiceDiscovery_002
TC_EVCC_DC_VTB_SDP_002	TC_SECC_CMN_VTB_ServiceDiscovery_003
TC_EVCC_DC_VTB_SDP_003	TC_SECC_AC_VTB_ServiceDiscovery_001
TC_EVCC_DC_VTB_SDP_004	TC_SECC_AC_VTB_ServiceDiscovery_002
TC_EVCC_DC_VTB_SDP_005	TC_SECC_DC_VTB_ServiceDiscovery_001
TC_EVCC_DC_VTB_SDP_006	TC_SECC_DC_VTB_ServiceDiscovery_002
TC_EVCC_DC_VTB_SDP_007	TC_SECC_CMN_VTB_ServiceDetail_003
TC_EVCC_DC_VTB_SDP_008	TC_SECC_CMN_VTB_ServiceDetail_004
TC_EVCC_DC_VTB_SDP_009	TC_SECC_CMN_VTB_ServiceDetail_005
TC_EVCC_DC_VTB_SDP_010	TC_SECC_CMN_VTB_ServiceDetail_006
TC_EVCC_DC_VTB_SDP_011	TC_SECC_CMN_VTB_ServiceDetail_007
TC_EVCC_DC_VTB_SDP_012	TC_SECC_CMN_VTB_ServiceDetail_008
TC_EVCC_DC_VTB_SDP_013	TC_SECC_CMN_VTB_PaymentServiceSelection_004
TC_EVCC_DC_VTB_SDP_014	TC_SECC_CMN_VTB_PaymentServiceSelection_005
TC_EVCC_DC_VTB_SDP_015	TC_SECC_CMN_VTB_PaymentServiceSelection_006
TC_EVCC_DC_VTB_SDP_016	TC_SECC_CMN_VTB_PaymentServiceSelection_007
TC_EVCC_CMN_VTB_SupportedAppProtocol_002	TC_SECC_CMN_VTB_PaymentServiceSelection_009
TC_EVCC_CMN_VTB_SupportedAppProtocol_003	TC_SECC_CMN_VTB_PaymentServiceSelection_010
TC_EVCC_CMN_VTB_SessionSetup_002	TC_SECC_CMN_VTB_CertificateInstallation_001
TC_EVCC_CMN_VTB_SessionSetup_003	TC_SECC_CMN_VTB_CertificateInstallation_002
TC_EVCC_CMN_VTB_SessionSetup_004	TC_SECC_CMN_VTB_CertificateInstallation_003
TC_EVCC_CMN_VTB_SessionSetup_005	TC_SECC_CMN_VTB_CertificateInstallation_004

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

TC_EVCC_CMN_VTB_SessionSetup_006	TC_SECC_CMN_VTB_CertificateInstallation_005
TC_EVCC_CMN_VTB_SessionSetup_007	TC_SECC_CMN_VTB_CertificateInstallation_006
TC_EVCC_AC_VTB_SessionSetup_001	TC_SECC_CMN_VTB_CertificateInstallation_007
TC_EVCC_AC_VTB_SessionSetup_002	TC_SECC_CMN_VTB_CertificateInstallation_008
TC_EVCC_AC_VTB_SessionSetup_003	TC_SECC_CMN_VTB_CertificateUpdate_001
TC_EVCC_AC_VTB_SessionSetup_004	TC_SECC_CMN_VTB_CertificateUpdate_002
TC_EVCC_DC_VTB_SessionSetup_001	TC_SECC_CMN_VTB_CertificateUpdate_003
TC_EVCC_DC_VTB_SessionSetup_002	TC_SECC_CMN_VTB_CertificateUpdate_004
TC_EVCC_DC_VTB_SessionSetup_003	TC_SECC_CMN_VTB_CertificateUpdate_005
TC_EVCC_DC_VTB_SessionSetup_004	TC_SECC_CMN_VTB_CertificateUpdate_006
TC_EVCC_CMN_VTB_V2GTPSessionSetup_002	TC_SECC_CMN_VTB_CertificateUpdate_008
TC_EVCC_CMN_VTB_V2GTPSessionSetup_005	TC_SECC_CMN_VTB_CertificateUpdate_009
TC_EVCC_CMN_VTB_ServiceDiscovery_002	TC_SECC_CMN_VTB_CertificateUpdate_010
TC_EVCC_CMN_VTB_ServiceDiscovery_003	TC_SECC_CMN_VTB_CertificateUpdate_011
TC_EVCC_CMN_VTB_ServiceDiscovery_004	TC_SECC_CMN_VTB_CertificateUpdate_012
TC_EVCC_CMN_VTB_ServiceDiscovery_005	TC_SECC_CMN_VTB_CertificateUpdate_013
TC_EVCC_CMN_VTB_ServiceDiscovery_006	TC_SECC_CMN_VTB_CertificateUpdate_014
TC_EVCC_CMN_VTB_ServiceDiscovery_007	TC_SECC_CMN_VTB_CertificateUpdate_015
TC_EVCC_CMN_VTB_ServiceDiscovery_008	TC_SECC_CMN_VTB_CertificateUpdate_016
TC_EVCC_CMN_VTB_ServiceDetail_003	TC_SECC_CMN_VTB_CertificateUpdate_017
TC_EVCC_CMN_VTB_ServiceDetail_004	TC_SECC_CMN_VTB_PaymentDetails_004
TC_EVCC_CMN_VTB_ServiceDetail_005	TC_SECC_CMN_VTB_PaymentDetails_005
TC_EVCC_CMN_VTB_ServiceDetail_006	TC_SECC_CMN_VTB_PaymentDetails_007
TC_EVCC_CMN_VTB_ServiceDetail_007	TC_SECC_CMN_VTB_PaymentDetails_008
TC_EVCC_CMN_VTB_ServiceDetail_008	TC_SECC_CMN_VTB_PaymentDetails_009
TC_EVCC_CMN_VTB_ServiceDetail_009	TC_SECC_CMN_VTB_PaymentDetails_010
TC_EVCC_CMN_VTB_ServiceDetail_010	TC_SECC_CMN_VTB_Authorization_001
TC_EVCC_CMN_VTB_ServiceDetail_011	TC_SECC_CMN_VTB_Authorization_003
TC_EVCC_CMN_VTB_ServiceDetail_012	TC_SECC_CMN_VTB_Authorization_004
TC_EVCC_CMN_VTB_ServiceDetail_013	TC_SECC_CMN_VTB_Authorization_005
TC_EVCC_CMN_VTB_ServiceDetail_014	TC_SECC_CMN_VTB_Authorization_006
TC_EVCC_CMN_VTB_ServiceDetail_015	TC_SECC_CMN_VTB_Authorization_007
TC_EVCC_CMN_VTB_ServiceDetail_017	TC_SECC_CMN_VTB_Authorization_010
TC_EVCC_CMN_VTB_ServiceDetail_018	TC_SECC_CMN_VTB_Authorization_011
TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_003	TC_SECC_CMN_VTB_Authorization_014

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_004	TC_SECC_CMN_VTB_Authorization_015
TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_005	TC_SECC_CMN_VTB_Authorization_016
TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_006	TC_SECC_CMN_VTB_Authorization_017
TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_007	TC_SECC_CMN_VTB_ChargeParameterDiscovery_001
TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_008	TC_SECC_AC_VTB_ChargeParameterDiscovery_003
TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_009	TC_SECC_AC_VTB_ChargeParameterDiscovery_004
TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_010	TC_SECC_AC_VTB_ChargeParameterDiscovery_005
TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_011	TC_SECC_AC_VTB_ChargeParameterDiscovery_007
TC_EVCC_CMN_VTB_ServiceDetailAndPaymentSelection_012	TC_SECC_AC_VTB_ChargeParameterDiscovery_011
TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_001	TC_SECC_AC_VTB_ChargeParameterDiscovery_014
TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_002	TC_SECC_AC_VTB_ChargeParameterDiscovery_017
TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_003	TC_SECC_AC_VTB_ChargeParameterDiscovery_018
TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_004	TC_SECC_AC_VTB_ChargeParameterDiscovery_019
TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_005	TC_SECC_AC_VTB_ChargeParameterDiscovery_020
TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_006	TC_SECC_AC_VTB_ChargeParameterDiscovery_021
TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_007	TC_SECC_AC_VTB_ChargeParameterDiscovery_022
TC_EVCC_AC_VTB_ServiceDetailAndPaymentSelection_008	TC_SECC_AC_VTB_ChargeParameterDiscovery_029
TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_001	TC_SECC_AC_VTB_ChargeParameterDiscovery_030
TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_002	TC_SECC_AC_VTB_ChargeParameterDiscovery_031
TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_003	TC_SECC_AC_VTB_ChargeParameterDiscovery_032
TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_004	TC_SECC_AC_VTB_ChargeParameterDiscovery_033
TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_005	TC_SECC_AC_VTB_ChargeParameterDiscovery_034
TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_006	TC_SECC_AC_VTB_ChargeParameterDiscovery_035
TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_007	TC_SECC_AC_VTB_ChargeParameterDiscovery_036
TC_EVCC_DC_VTB_ServiceDetailAndPaymentSelection_008	TC_SECC_DC_VTB_ChargeParameterDiscovery_002
TC_EVCC_AC_VTB_PaymentServiceSelection_002	TC_SECC_DC_VTB_ChargeParameterDiscovery_003
TC_EVCC_AC_VTB_PaymentServiceSelection_003	TC_SECC_DC_VTB_ChargeParameterDiscovery_004
TC_EVCC_AC_VTB_PaymentServiceSelection_004	TC_SECC_DC_VTB_ChargeParameterDiscovery_007
TC_EVCC_AC_VTB_PaymentServiceSelection_005	TC_SECC_DC_VTB_ChargeParameterDiscovery_011
TC_EVCC_AC_VTB_PaymentServiceSelection_006	TC_SECC_DC_VTB_ChargeParameterDiscovery_014
TC_EVCC_AC_VTB_PaymentServiceSelection_007	TC_SECC_DC_VTB_ChargeParameterDiscovery_017
TC_EVCC_AC_VTB_PaymentServiceSelection_008	TC_SECC_DC_VTB_ChargeParameterDiscovery_018
TC_EVCC_AC_VTB_PaymentServiceSelection_009	TC_SECC_DC_VTB_ChargeParameterDiscovery_019
TC_EVCC_AC_VTB_PaymentServiceSelection_010	TC_SECC_DC_VTB_ChargeParameterDiscovery_020
TC_EVCC_AC_VTB_PaymentServiceSelection_011	TC_SECC_DC_VTB_ChargeParameterDiscovery_021

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

TC_EVCC_AC_VTB_PaymentServiceSelection_012	TC_SECC_DC_VTB_ChargeParameterDiscovery_022
TC_EVCC_AC_VTB_PaymentServiceSelection_013	TC_SECC_DC_VTB_ChargeParameterDiscovery_029
TC_EVCC_AC_VTB_PaymentServiceSelection_014	TC_SECC_DC_VTB_ChargeParameterDiscovery_030
TC_EVCC_AC_VTB_PaymentServiceSelection_015	TC_SECC_DC_VTB_ChargeParameterDiscovery_031
TC_EVCC_AC_VTB_PaymentServiceSelection_016	TC_SECC_DC_VTB_ChargeParameterDiscovery_032
TC_EVCC_AC_VTB_PaymentServiceSelection_017	TC_SECC_DC_VTB_ChargeParameterDiscovery_033
TC_EVCC_DC_VTB_PaymentServiceSelection_001	TC_SECC_DC_VTB_ChargeParameterDiscovery_034
TC_EVCC_DC_VTB_PaymentServiceSelection_002	TC_SECC_DC_VTB_ChargeParameterDiscovery_035
TC_EVCC_DC_VTB_PaymentServiceSelection_003	TC_SECC_DC_VTB_ChargeParameterDiscovery_036
TC_EVCC_DC_VTB_PaymentServiceSelection_004	TC_SECC_AC_VTB_PowerDelivery_006
TC_EVCC_DC_VTB_PaymentServiceSelection_005	TC_SECC_AC_VTB_PowerDelivery_007
TC_EVCC_DC_VTB_PaymentServiceSelection_006	TC_SECC_AC_VTB_PowerDelivery_008
TC_EVCC_DC_VTB_PaymentServiceSelection_007	TC_SECC_AC_VTB_PowerDelivery_009
TC_EVCC_DC_VTB_PaymentServiceSelection_008	TC_SECC_AC_VTB_PowerDelivery_010
TC_EVCC_DC_VTB_PaymentServiceSelection_009	TC_SECC_AC_VTB_PowerDelivery_011
TC_EVCC_DC_VTB_PaymentServiceSelection_010	TC_SECC_AC_VTB_PowerDelivery_014
TC_EVCC_DC_VTB_PaymentServiceSelection_011	TC_SECC_AC_VTB_ChargingStatus_003
TC_EVCC_DC_VTB_PaymentServiceSelection_012	TC_SECC_AC_VTB_ChargingStatus_004
TC_EVCC_DC_VTB_PaymentServiceSelection_013	TC_SECC_AC_VTB_MeteringReceipt_002
TC_EVCC_DC_VTB_PaymentServiceSelection_014	TC_SECC_AC_VTB_MeteringReceipt_003
TC_EVCC_DC_VTB_PaymentServiceSelection_015	TC_SECC_AC_VTB_MeteringReceipt_004
TC_EVCC_DC_VTB_PaymentServiceSelection_016	TC_SECC_AC_VTB_MeteringReceipt_006
TC_EVCC_CMN_VTB_CertificateInstallation_002	TC_SECC_AC_VTB_MeteringReceipt_007
TC_EVCC_CMN_VTB_CertificateInstallation_003	TC_SECC_AC_VTB_SessionStop_002
TC_EVCC_CMN_VTB_CertificateInstallation_004	TC_SECC_AC_VTB_SessionStop_003
TC_EVCC_CMN_VTB_CertificateInstallation_005	TC_SECC_DC_VTB_CableCheck_003
TC_EVCC_CMN_VTB_CertificateInstallation_006	TC_SECC_DC_VTB_CableCheck_004
TC_EVCC_CMN_VTB_CertificateInstallation_007	TC_SECC_DC_VTB_CableCheck_006
TC_EVCC_CMN_VTB_CertificateInstallation_008	TC_SECC_DC_VTB_CableCheck_008
TC_EVCC_CMN_VTB_CertificateInstallation_009	TC_SECC_DC_VTB_PreCharge_002
TC_EVCC_CMN_VTB_CertificateInstallation_010	TC_SECC_DC_VTB_PreCharge_003
TC_EVCC_CMN_VTB_CertificateInstallation_011	TC_SECC_DC_VTB_PowerDelivery_003
TC_EVCC_CMN_VTB_CertificateInstallation_012	TC_SECC_DC_VTB_PowerDelivery_004
TC_EVCC_CMN_VTB_CertificateUpdate_002	TC_SECC_DC_VTB_PowerDelivery_005
TC_EVCC_CMN_VTB_CertificateUpdate_003	TC_SECC_DC_VTB_PowerDelivery_006

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

TC_EVCC_CMN_VTB_CertificateUpdate_004	TC_SECC_DC_VTB_PowerDelivery_007
TC_EVCC_CMN_VTB_CertificateUpdate_005	TC_SECC_DC_VTB_PowerDelivery_008
TC_EVCC_CMN_VTB_CertificateUpdate_006	TC_SECC_DC_VTB_PowerDelivery_009
TC_EVCC_CMN_VTB_CertificateUpdate_007	TC_SECC_DC_VTB_PowerDelivery_013
TC_EVCC_CMN_VTB_CertificateUpdate_008	TC_SECC_DC_VTB_CurrentDemand_003
TC_EVCC_CMN_VTB_CertificateUpdate_009	TC_SECC_DC_VTB_CurrentDemand_004
TC_EVCC_CMN_VTB_CertificateUpdate_010	TC_SECC_DC_VTB_MeteringReceipt_002
TC_EVCC_CMN_VTB_CertificateUpdate_011	TC_SECC_DC_VTB_MeteringReceipt_003
TC_EVCC_CMN_VTB_CertificateUpdate_012	TC_SECC_DC_VTB_MeteringReceipt_004
TC_EVCC_CMN_VTB_CertificateUpdate_013	TC_SECC_DC_VTB_MeteringReceipt_006
TC_EVCC_CMN_VTB_CertificateUpdate_014	TC_SECC_DC_VTB_MeteringReceipt_007
TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_002	TC_SECC_DC_VTB_WeldingDetection_002
TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_003	TC_SECC_DC_VTB_WeldingDetection_003
TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_004	TC_SECC_DC_VTB_WeldingDetection_004
TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_005	TC_SECC_DC_VTB_WeldingDetection_005
TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_006	TC_SECC_DC_VTB_SessionStop_003
TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_007	TC_SECC_DC_VTB_SessionStop_004
TC_EVCC_CMN_VTB_CertificateHandlingAndPaymentDetails_008	TC_SECC_DC_VTB_SessionStop_005
TC_EVCC_CMN_VTB_Authorization_003	TC_SECC_DC_VTB_SessionStop_006
TC_EVCC_CMN_VTB_Authorization_004	TC_SECC_DC_VTB_SessionStop_007
TC_EVCC_CMN_VTB_Authorization_005	TC_SECC_DC_VTB_SessionStop_008
TC_EVCC_CMN_VTB_Authorization_006	TC_SECC_DC_VTB_SessionStop_009
TC_EVCC_CMN_VTB_Authorization_007	
TC_EVCC_CMN_VTB_Authorization_008	
TC_EVCC_CMN_VTB_Authorization_009	
TC_EVCC_CMN_VTB_Authorization_012	
TC_EVCC_CMN_VTB_Authorization_013	
TC_EVCC_CMN_VTB_Authorization_014	
TC_EVCC_CMN_VTB_ChargeParameterDiscovery_001	
TC_EVCC_CMN_VTB_ChargeParameterDiscovery_002	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_002	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_003	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_004	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_005	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_006	

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

TC_EVCC_AC_VTB_ChargeParameterDiscovery_007	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_009	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_010	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_015	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_016	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_017	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_018	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_019	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_020	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_021	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_022	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_023	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_024	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_025	
TC_EVCC_AC_VTB_ChargeParameterDiscovery_026	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_002	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_003	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_004	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_005	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_006	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_007	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_009	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_010	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_015	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_016	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_017	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_018	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_019	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_020	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_021	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_022	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_023	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_024	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_025	
TC_EVCC_DC_VTB_ChargeParameterDiscovery_026	

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

TC_EVCC_AC_VTB_PowerDelivery_002	
TC_EVCC_AC_VTB_PowerDelivery_003	
TC_EVCC_AC_VTB_PowerDelivery_004	
TC_EVCC_AC_VTB_PowerDelivery_005	
TC_EVCC_AC_VTB_PowerDelivery_006	
TC_EVCC_AC_VTB_PowerDelivery_007	
TC_EVCC_AC_VTB_PowerDelivery_008	
TC_EVCC_AC_VTB_PowerDelivery_009	
TC_EVCC_AC_VTB_PowerDelivery_010	
TC_EVCC_AC_VTB_PowerDelivery_011	
TC_EVCC_AC_VTB_PowerDelivery_012	
TC_EVCC_AC_VTB_PowerDelivery_013	
TC_EVCC_AC_VTB_PowerDelivery_014	
TC_EVCC_AC_VTB_PowerDelivery_016	
TC_EVCC_AC_VTB_PowerDelivery_017	
TC_EVCC_AC_VTB_PowerDelivery_022	
TC_EVCC_AC_VTB_PowerDelivery_023	
TC_EVCC_AC_VTB_PowerDelivery_024	
TC_EVCC_AC_VTB_PowerDelivery_025	
TC_EVCC_AC_VTB_PowerDelivery_026	
TC_EVCC_AC_VTB_PowerDelivery_027	
TC_EVCC_AC_VTB_PowerDelivery_028	
TC_EVCC_AC_VTB_PowerDelivery_029	
TC_EVCC_AC_VTB_PowerDelivery_030	
TC_EVCC_AC_VTB_PowerDelivery_031	
TC_EVCC_AC_VTB_PowerDelivery_032	
TC_EVCC_AC_VTB_PowerDelivery_033	
TC_EVCC_AC_VTB_ChargingStatus_002	
TC_EVCC_AC_VTB_ChargingStatus_003	
TC_EVCC_AC_VTB_ChargingStatus_004	
TC_EVCC_AC_VTB_ChargingStatus_005	
TC_EVCC_AC_VTB_ChargingStatus_006	
TC_EVCC_AC_VTB_ChargingStatus_007	
TC_EVCC_AC_VTB_ChargingStatus_008	
TC_EVCC_DC_VTB_CableCheck_002	

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

TC_EVCC_DC_VTB_CableCheck_003	
TC_EVCC_DC_VTB_CableCheck_004	
TC_EVCC_DC_VTB_CableCheck_005	
TC_EVCC_DC_VTB_CableCheck_006	
TC_EVCC_DC_VTB_CableCheck_007	
TC_EVCC_DC_VTB_CableCheck_008	
TC_EVCC_DC_VTB_CableCheck_009	
TC_EVCC_DC_VTB_PreCharge_002	
TC_EVCC_DC_VTB_PreCharge_003	
TC_EVCC_DC_VTB_PreCharge_004	
TC_EVCC_DC_VTB_PreCharge_005	
TC_EVCC_DC_VTB_PreCharge_006	
TC_EVCC_DC_VTB_PreCharge_007	
TC_EVCC_DC_VTB_PreCharge_008	
TC_EVCC_DC_VTB_PreCharge_009	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_002	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_003	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_004	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_005	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_006	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_007	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_008	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_009	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_010	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_011	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_012	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_013	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_014	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_017	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_018	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_019	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_020	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_021	
TC_EVCC_DC_VTB_PreChargeOrPowerDelivery_022	
TC_EVCC_DC_VTB_CurrentDemand_002	

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

TC_EVCC_DC_VTB_CurrentDemand_003	
TC_EVCC_DC_VTB_CurrentDemand_004	
TC_EVCC_DC_VTB_CurrentDemand_005	
TC_EVCC_DC_VTB_CurrentDemand_006	
TC_EVCC_DC_VTB_CurrentDemand_007	
TC_EVCC_DC_VTB_CurrentDemand_008	
TC_EVCC_AC_VTB_MeteringReceipt_002	
TC_EVCC_AC_VTB_MeteringReceipt_003	
TC_EVCC_AC_VTB_MeteringReceipt_004	
TC_EVCC_AC_VTB_MeteringReceipt_005	
TC_EVCC_AC_VTB_MeteringReceipt_006	
TC_EVCC_AC_VTB_MeteringReceipt_007	
TC_EVCC_AC_VTB_MeteringReceipt_010	
TC_EVCC_AC_VTB_MeteringReceipt_011	
TC_EVCC_AC_VTB_MeteringReceipt_012	
TC_EVCC_DC_VTB_MeteringReceipt_002	
TC_EVCC_DC_VTB_MeteringReceipt_003	
TC_EVCC_DC_VTB_MeteringReceipt_004	
TC_EVCC_DC_VTB_MeteringReceipt_005	
TC_EVCC_DC_VTB_MeteringReceipt_006	
TC_EVCC_DC_VTB_MeteringReceipt_007	
TC_EVCC_DC_VTB_MeteringReceipt_010	
TC_EVCC_DC_VTB_MeteringReceipt_011	
TC_EVCC_DC_VTB_MeteringReceipt_012	
TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_001	
TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_002	
TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_003	
TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_004	
TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_005	
TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_006	
TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_007	
TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_010	
TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_011	
TC_EVCC_AC_VTB_ChargingLoopAcOrPowerDelivery_012	
TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_001	

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_002	
TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_003	
TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_004	
TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_005	
TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_006	
TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_007	
TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_010	
TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_011	
TC_EVCC_DC_VTB_ChargingLoopDcOrPowerDelivery_012	
TC_EVCC_DC_VTB_WeldingDetection_002	
TC_EVCC_DC_VTB_WeldingDetection_003	
TC_EVCC_DC_VTB_WeldingDetection_004	
TC_EVCC_DC_VTB_WeldingDetection_005	
TC_EVCC_DC_VTB_WeldingDetection_006	
TC_EVCC_DC_VTB_WeldingDetection_007	
TC_EVCC_DC_VTB_WeldingDetection_008	
TC_EVCC_DC_VTB_WeldingDetection_009	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_002	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_003	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_004	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_005	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_006	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_012	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_013	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_014	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_015	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_016	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_017	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_018	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_019	
TC_EVCC_DC_VTB_WeldingDetectionOrSessionStop_020	
TC_EVCC_AC_VTB_SessionStop_002	
TC_EVCC_AC_VTB_SessionStop_003	
TC_EVCC_AC_VTB_SessionStop_004	
TC_EVCC_AC_VTB_SessionStop_005	

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

TC_EVCC_AC_VTB_SessionStop_006	
TC_EVCC_AC_VTB_SessionStop_012	
TC_EVCC_AC_VTB_SessionStop_013	
TC_EVCC_AC_VTB_SessionStop_014	
TC_EVCC_AC_VTB_SessionStop_015	
TC_EVCC_AC_VTB_SessionStop_016	
TC_EVCC_AC_VTB_SessionStop_017	
TC_EVCC_AC_VTB_SessionStop_018	
TC_EVCC_AC_VTB_SessionStop_019	
TC_EVCC_AC_VTB_SessionStop_020	
TC_EVCC_DC_VTB_SessionStop_013	
TC_EVCC_DC_VTB_SessionStop_014	
TC_EVCC_DC_VTB_SessionStop_015	
TC_EVCC_DC_VTB_SessionStop_016	
TC_EVCC_DC_VTB_SessionStop_017	
TC_EVCC_DC_VTB_SessionStop_018	
TC_EVCC_DC_VTB_SessionStop_019	
TC_EVCC_DC_VTB_SessionStop_020	
TC_EVCC_DC_VTB_SessionStop_021	
TC_EVCC_DC_VTB_SessionStop_022	
TC_EVCC_DC_VTB_SessionStop_023	
TC_EVCC_DC_VTB_SessionStop_024	
TC_EVCC_DC_VTB_SessionStop_025	
TC_EVCC_DC_VTB_SessionStop_026	
TC_EVCC_DC_VTB_SessionStop_027	
TC_EVCC_DC_VTB_SessionStop_028	
TC_EVCC_DC_VTB_SessionStop_029	
TC_EVCC_DC_VTB_SessionStop_030	

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb

Datum / Uhrzeit des Ausdrucks: 2025-12-02, 12:53:47

Printed copies are uncontrolled

Bibliography

[1] ISO 15118-1, *Road vehicles — Vehicle to grid communication interface — Part 1: General information and use-case definition*

[2] ITU-T X.292, *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications — The Tree and Tabular Combined Notation (TTCN) (May 2002)*

[3] ETSI ES 201 873-1 V4.6.1, *TTCN-3: Core Language (June 2014)*

[4] ETSI ES 201 873-9 V4.6.1, *TTCN-3: Using XML schema with TTCN-3 (June 2015)*

[5] ITU-T X.290, OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications — General concepts (April 1995)

[6] IEC FDIS 61851-23:2023, *Electric vehicle conductive charging system — Part 23: DC electric vehicle charging station (Edition 2)*

© ISO 2024 – All rights reserved

Firmenname: Hochschule für Technik, Wirtschaft und Kultur Benutzername: _ip_user_c8563dc5-739d-4631-bf84-0b2da5471fdb