VOLKSWAGEN

AKTIENGESELLSCHAFT

CONFIDENTIAL
VERTRAULICH

UNECE Softwareupdate - Allgemeine Diagnose-anforderungen

Ergänzungsspezifikation für Steuergeräte

Technische Entwicklung, Funktionslastenheft: LAH.DUM.905.E

Autor	Peter-Michael Hofmann (EESN/4), Dora Aranyi (I/EE-87), Dr. Markus Koch (EEY3)
Abt./OE	EESN/4, I/EE-87, EEY3
Telefon	+49-5361-9-78398, +49-841-89-91654, +49-711-911-83633
E-Mail	peter.hofmann@volkswagen.de, dora.aranyi@audi.de, markus.koch@porsche.de

Erstausgabe	22.11.2019
Änderungsstand	03.12.2019
Lastenheftversion	1.0.1
Baseline	2.8 ( )

Inhaltsverzeichnis

1 Allgemeines

1.1 Zweck

[I: F-LAH_RxSWIN-7]

Dieses Dokument beschreibt technische Anforderungen an Onboard- sowie Offboard-Systeme zur Einhaltung der "UNECE Regulierung zu Software Updates" der "Working Party on Automated/Autonomous and Connected Vehicles (GRVA)" aus der WP.29 ("Regulation on uniform provisions concerning the approval of software update processes").

[I: F-LAH_RxSWIN-712]

Jede Software zur Umsetzung von Fahrzeugfunktionen muss entsprechend dem Software Update Management System entwickelt werden. Zusätzlich müssen typgenehmigungspflichtige Fahrzeugfunktionen eine regulierungsbezogene Software Identifikationsnummer (RxSWIN) erhalten. Dies betrifft beispielsweise UNECE- oder GB/T-Regulierungen. Diese Anforderungen sind verpflichtend zu erfüllen, um eine Typgenehmigung zu erhalten.

[I: F-LAH_RxSWIN-433]

Seitens der Diagnose ergeben sich durch die UNECE Regulierung Software Update folgende Anforderungen:

[I: F-LAH_RxSWIN-732]

- Authentisches Einbringen von Software (Instruction Code and Data)
  Referenz: UNECE Regulierung zu Software Updates-7.2.1.1. - The authenticity and integrity of software updates shall be protected to reasonably prevent their compromise and reasonably prevent invalid updates.
  Umsetzung: Nutzung von Flashdatensicherheit oder ein alternatives, mindestens gleichwertiges Absicherungsverfahren und Schutz der Fahrzeugdiagnose

[I: F-LAH_RxSWIN-733]

- Einführen von RxSWIN für typgenehmigungspflichtige Fahrzeugfunktionen
  Referenz: UNECE Regulierung zu Software Updates-2.2. - “Regulation X Software Identification Number (RXSWIN)” means a dedicated identifier, defined by the vehicle manufacturer, representing information about the type approval relevant software of the Electronic Control System contributing to the Regulation N° x type approval relevant characteristics of the vehicle.
  Umsetzung: Einführung eines RxSWIN-DataIdentifier im Gateway

[I: F-LAH_RxSWIN-734]

- Einführen eines Integritätsmerkmals von Software (Instruction Code and Data)
  Referenz: UNECE Regulierung zu Software Updates-7.1.2.3. - For every RxSWIN, there shall be documentation describing the software relevant to the RxSWIN of the vehicle type before and after an update. This shall include information of the software versions and their integrity validation data for all relevant software for each RxSWIN.
  Umsetzung: Einführung von Integrity Validation Data für Programm und Daten

[I: F-LAH_RxSWIN-552]

Bei Fahrzeugen mit einer neuen Typgenehmigung für den Markt Japan ab Januar 2022 und für den Markt EU ab Mai 2022 dürfen ohne die technischen Voraussetzungen (RxSWIN) und eine fehlende SUMS-Zertifizierung keine Softwareupdates durchgeführt werden. Aufgrund der deshalb nicht möglichen Software Updates- und Cyber Security-Typprüfung ist eine Typgenehmigung des Fahrzeugs nicht möglich! Dies wird für den Markt Japan ab Januar 2024 und für den Markt EU ab Mai 2024 auf alle Bestandsfahrzeuge ausgeweitet.

[I: F-LAH_RxSWIN-430]

Die Steuergeräte werden entsprechend ihrer Plattformzugehörigkeit eingeteilt. Dabei wird zwischen Bestandsplattformen und neuen Plattformen unterschieden:

[I: F-LAH_RxSWIN-827]

- Bestandsplattformen sind z. B. MLBevo Gen2, MQB, MEB

[I: F-LAH_RxSWIN-828]

- Neue Plattformen sind: MEB 1.1 Advanced, E³ 1.2 Premium sowie E³ 2.0 und alle folgenden Plattformen

[I: F-LAH_RxSWIN-432]

Für Fahrzeugprojekte mit einer Typgenehmigung ab Januar 2022 gelten folgende Anforderungen:

[I: F-LAH_RxSWIN-825]

Alle programmier- oder konfigurierbare Steuergeräte:

- ECU programming data security (EPDS) für Software
- Schutz der Fahrzeugdiagnose (SFD-Ende-zu-Ende-Absicherung) für Daten
    - Ausnahme DK2-System: Absicherung erfolgt über das übergeordnete DK4-low-System
- Integrity Validation Data für Software und Daten

[I: F-LAH_RxSWIN-826]

Gateway-Steuergerät zusätzlich:

- Umsetzung Diagnosefilter mit SFD-Zugriffsschutz
- RxSWIN mit SFD-Ende-zu-Ende-Absicherung

[I: F-LAH_RxSWIN-746]

Für alle Fahrzeugprojekte (auch Bestandsplattformen) gelten ab Januar 2024 grundsätzlich die gleichen Anforderungen:

- Eine Abstimmung über die Anforderungen für Bestandsplattformen findet in der GRVA bis etwa Ende Februar 2020 statt. Nach dem Vorliegen der endgültigen Regulierung wird das RxSWIN-FKT-LAH ggf. angepasst.

1.2 Abkürzungen und Begriffe

[I: F-LAH_RxSWIN-21]

Tabelle 1-1 Abkürzungen und Begriffe

Abkürzung oder Begriff	Bezeichnung	Bedeutung
BoM	Bill of Material	Bauzustandsdokumenatation
BSB	Bauteil-Sachbearbeiter	Rollenbezeichnung d. AUDI AG, gleichzusetzen mit "BTV - Bauteil-Verantwortlicher" d. Volkswagen AG
C	Conditional	Cvt.=C: In Abhängigkeit von bestimmten Bedingungen zu senden/zu implementieren.
Cvt.	Convention	Implementierungsregeln und Konventionen, die für Parameter eines Services gelten.
EPDS	ECU programming data security	Generischer Begriff für einen Security-Mechanismus der die Authentizität und Integrität eines Steuergeräte-Programmstands über Diagnose feststellt. Dies kann durch die Umsetzung von Flashdatensicherheit (FDS) oder ein alternatives, mindestens gleichwertiges Absicherungsverfahren erfolgen, welches mit der E/E-Security-Abteilung des Auftraggebers abgestimmt ist.
FDS	Flashdatensicherheit	Kryptografisches Verfahren zur Absicherung der Flashdaten.
GB/T	GB steht für Guobiao, chinesisch für „Nationaler Standard“	Grundlage für den Produkttest, den das Produkt im Zuge der CCC-Zertifizierung durchlaufen muss.
IVD	Integrity Validation Data	Hashwerte über Software (Instruction-Code) und Konfigurationsdaten (Data)
M	Mandatory	Cvt.=M: Verbindlich für die Applikations-Software zu implementieren bzw. immer zu senden (engl. für befehlend).
NoImp	No implementation	Eine Implementierung darf für VOLKSWAGEN AG beauftragte Server nicht erfolgen.
RxSWIN	Regulierungsbezogene Software Identifikation	Software Identifikationsnummer pro betroffener Regulierung.
SFD	Schutz der Fahrzeugdiagnose	Kryptografisches Verfahren zur Absicherung a) des Zugriffs auf Diagnoseobjekte (SFD-Zugriffsschutz) bzw. b) von Diagnoseinhalten (SFD-Ende-zu-Ende-Absicherung).
SUMS	Software Update Management System	Stellt die Einhaltung von gesetzlichen Anforderungen bzgl. der Bereitstellung von Software Updates durch entsprechende Prozesse beim Hersteller sicher.
SW	Software	Nach UNECE SU ist mit Software der ausführbarer Code und die Daten (Instruction Code and Data) gemeint.
U	User-Optional	Cvt.= NoImp: Eine Implementierung darf für VOLKSWAGEN AG beauftragte Server nicht erfolgen.
UNECE	United Nations Economic Commission for Europe	Wirtschaftskommission für Europa der Vereinten Nationen
UNECE R SU	UNECE Regulierung zu Software Update	Titel: "Regulation on uniform provisions concerning the approval of software update processes".
ZDC	Zieldaten-Container	XML-Datei, die alle Parameter der Varianten eines Servers enthält, die über PR-Nummern ausgewählt werden.
zGW	Zentrales Vernetzungs-Gateway	Steuergerät mit zentralem Diagnosezugang, z.B. Gateway, ICAS1, HCP5.
VOLKSWAGEN AG-reserved		Die Verwendung ist für zukünftige Anwendungen der VOLKSWAGEN AG reserviert.

1.3 Gültigkeitsbereich

[allg. Anf.: F-LAH_RxSWIN-9]

Dieses Dokument ist gültig für alle programmier- und konfigurierbaren Steuergeräte in Bestandsplattformen sowie in neuen Plattformen.

[allg. Anf.: F-LAH_RxSWIN-550]

Bei abweichenden Anforderungen zwischen diesem Dokument und den anderen Diagnose-Querschnittslastenheften gelten die Anforderungen aus diesem Lastenheft.

[I: F-LAH_RxSWIN-554]

In den nächsten Releases der Diagnose-Querschnittslastenheften werden die Anforderungen aus diesem Lastenheft mit den Anforderungen aus den anderen Dokumenten zusammengeführt.

[I: F-LAH_RxSWIN-521]

Die Farbkennzeichnung der Tabellen dient nur der Lesbarkeit.

[allg. Anf.: F-LAH_RxSWIN-918]

Die als "Prozess-Anforderung" gekennzeichneten Anforderungen sind im Rahmen der Steuergeräteentwicklung nicht zu berücksichtigen.

2 Anforderungen an Steuergeräte

[I: F-LAH_RxSWIN-917]

Tabelle 2-1 Übersicht der Anforderungen aus Sicht Diagnoseklasse

	Zentrales Gateway	DK4(V)	DK3(V)	DK2F(V)	DK2(V)	DK1(V)/0
RxSWIN DataIdentifier	X	NoImp	NoImp	NoImp	NoImp	NoImp
Diagnostic filter (OBD-port)	X	NoImp	NoImp	NoImp	NoImp	NoImp
ECU programming data security (EPDS)	X	X	X 3)	X	NoImp	NoImp
Protection of vehicle diagnosis (PVD/SFD) **- SFD Authentication** **- SFD End2End**	X	X	X 1)	NoImp 2)	NoImp 2)	NoImp
Integrity validation data (IVD)	X	X 3)	X 3)	X 3)	X 3)	NoImp

[I: F-LAH_RxSWIN-922]

x = beinhaltet

[I: F-LAH_RxSWIN-919]

1) = Wenn keine Konfiguration mittels ZDC in der Applikation möglich ist, dann ist der Einsatz von SFD abhängig von der Risiko- /Securityanalyse.

[I: F-LAH_RxSWIN-923]

2) = erhalten die Daten ohne Signatur vom DK4

[I: F-LAH_RxSWIN-943]

3) = nur bei bedatbaren (programmier- und/oder konfigurierbaren) Diagnose-Servern

[I: F-LAH_RxSWIN-920]

Alle Anforderungen gelten auch für Carry Over Parts.

[I: F-LAH_RxSWIN-921]

Alle Anforderungen sind Mindestanforderungen aus der UNECE Softwareupdate. Durch die Risiko-/Securityanalyse können sich höhere Anforderungen ergeben.

2.1 DK0/DK1-Systeme

[I: F-LAH_RxSWIN-709]

DK0/DK1-Systeme sind von den Anforderungen in diesem Dokument nicht betroffen, da diese nicht bedatbar sind.

2.2 DK2/DK2F/DK2FV-Systeme

[I: F-LAH_RxSWIN-941]

Nicht bedatbare (weder programmier- noch konfigurierbare) DK2/DK2F/DK2FV-Systeme sind von diesem Dokument nicht betroffen.

2.2.1 Programmierbare DK2F/DK2FV-Systeme

[allg. Anf.: F-LAH_RxSWIN-850]

Programmierbare DK2F/DK2FV-Systeme müssen

[allg. Anf.: F-LAH_RxSWIN-754]

- ECU programming data security (EPDS) unterstützen.

[allg. Anf.: F-LAH_RxSWIN-847]

- Integrity Validation Data für die Programmierung unterstützen. Siehe Kapitel "Integrity Validation Data".

2.2.2 Konfigurierbare DK2/DK2F/DK2FV-Systeme

[allg. Anf.: F-LAH_RxSWIN-851]

Konfigurierbare DK2/DK2F/DK2FV-Systeme

[allg. Anf.: F-LAH_RxSWIN-755]

- werden per ZDC konfiguriert.

[allg. Anf.: F-LAH_RxSWIN-756]

- erhalten Daten ohne Signatur vom DK4-low-System. Das DK4-low-System führt die SFD-E2E-Signaturprüfung für das DK2/DK2F/DK2FV-System durch.

[allg. Anf.: F-LAH_RxSWIN-757]

- müssen Integrity Validation Data für die Konfiguration unterstützen. Siehe Kapitel "Integrity Validation Data".

2.3 DK3/DK3V/DK4/DK4V-Systeme und SWCL

[I: F-LAH_RxSWIN-942]

Nicht bedatbare (weder programmier- noch konfigurierbare) DK3/DK3V/DK4/DK4V-Systeme und SWCL sind von diesem Dokument nicht betroffen.

2.3.1 Programmierbare DK3/DK3V/DK4/DK4V-Systeme und SWCL

[allg. Anf.: F-LAH_RxSWIN-852]

Programmierbare DK3/DK3V/DK4/DK4V-Systeme und SWCL müssen

[allg. Anf.: F-LAH_RxSWIN-759]

- ECU programming data security (EPDS) unterstützen.

[allg. Anf.: F-LAH_RxSWIN-849]

- Integrity Validation Data für die Programmierung unterstützen. Siehe Kapitel "Integrity Validation Data".

2.3.2 Konfigurierbare DK3/DK3V/DK4/DK4V-Systeme und SWCL

[allg. Anf.: F-LAH_RxSWIN-853]

Konfigurierbare DK3/DK3V/DK4/DK4V-Systeme und SWCL

[allg. Anf.: F-LAH_RxSWIN-760]

- werden per ZDC konfiguriert.

[allg. Anf.: F-LAH_RxSWIN-761]

- müssen SFD-Ende-zu-Ende-Absicherung (SFD-E2E) unterstützen, wenn sie in der Applikation konfigurierbar sind. Siehe Kapitel "SFD-E2E-Absicherung".

[allg. Anf.: F-LAH_RxSWIN-762]

- müssen Integrity Validation Data für die Konfiguration unterstützen. Siehe Kapitel "Integrity Validation Data".

2.3.3 Zusätzliche Anforderungen an DK4-low/DK4V-low-Systeme

2.3.3.1 Sammel-DataIdentifier

[allg. Anf.: F-LAH_RxSWIN-804]

DK4-low/DK4V-low-Systeme müssen:

[allg. Anf.: F-LAH_RxSWIN-506]

- den Sammel-DataIdentifier für Integrity Validation Data der Konfiguration von DK2/DK2F/DK2FV-Systemen ausgegeben. Siehe Kapitel "Integrity Validation Data".

[allg. Anf.: F-LAH_RxSWIN-788]

- den Sammel-DataIdentifier für Integrity Validation Data der Programmierung von DK2F/DK2FV-Systemen ausgegeben. Siehe Kapitel "Integrity Validation Data".

2.3.3.2 Routingmechanismus im DK4-low/DK4V-low-System für ein DK2F/DK2FV-System

[I: F-LAH_RxSWIN-59]

Die Aktivierung des Routingmechanismus im DK4-low/DK4V-low-Systems wird gemäß Dokument /3/ durch den Service CommunicationControl (28hex) mit dem ControlType [0x04 - enableRxAndDisableTxWithEnhancedAddressInformation] gestartet.

[allg. Anf.: F-LAH_RxSWIN-57]

Ein DK4-low/DK4V-low-System muss den Service CommunicationControl (28hex) für den ControlType [0x04-EnableRxAndDisableTxWithEnhancedAddressInformation] mit SFD-Zugriffsschutz schützen.

[allg. Anf.: F-LAH_RxSWIN-78]

Für den ControlType [0x04 - enableRxAndDisableTxWithEnhancedAddressInformation] ist der SFD-Zugriffschutz mit der Rolle "Basic" umzusetzen.

[allg. Anf.: F-LAH_RxSWIN-79]

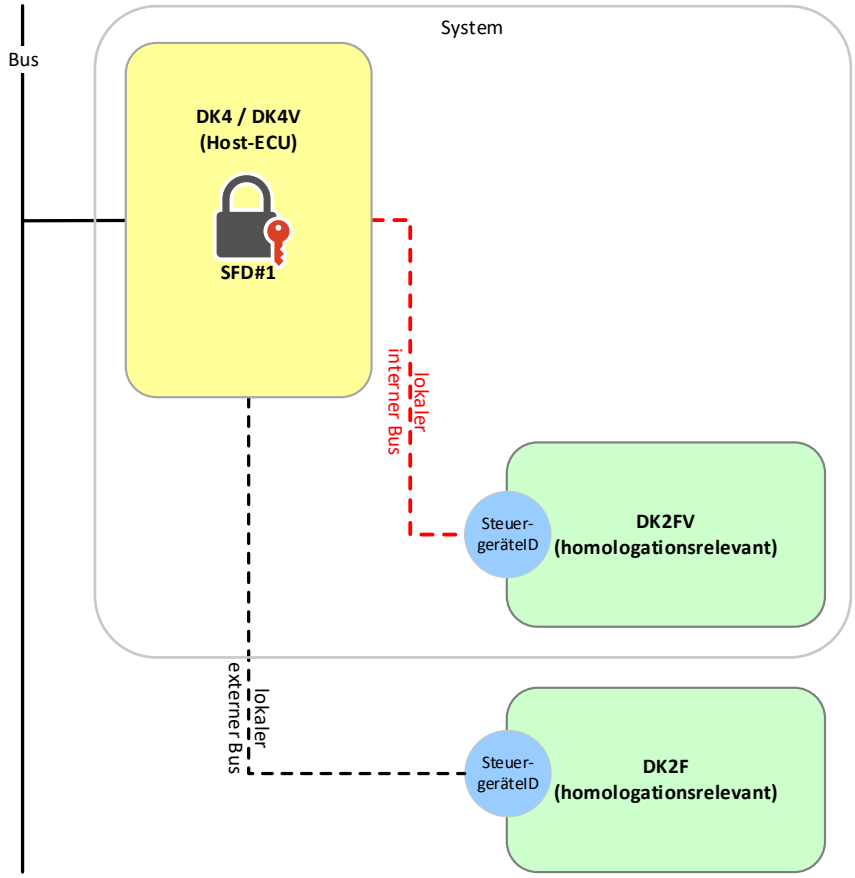
Entgegen der Anforderung 80127-3047 bis v5.8 gilt:
Für alle DK2F-Systeme ist es explizit verboten im DK4-low-System das Routing der Diagnosekommunikation im Normalbetrieb permanent umzusetzen.

[allg. Anf.: F-LAH_RxSWIN-80]

Wenn ein DK4-low/DK4V-low-System und ein DK2F-System am gleichen Bussegment angeschlossen sind und über dieses Bussegment für beide Systeme Diagnosekommunikation betrieben wird, dann müssen die Security-Aspekte mit der Diagnosefachabteilung und der E/E-Security-Abteilung abgestimmt werden.

[I: F-LAH_RxSWIN-104]

Abbildung 2-1 RoutingActivation DK4-low, geschützt über SFD-Zugriffsschutz



2.3.4 Zusätzliche Anforderungen an Gateway-Steuergeräte

[allg. Anf.: F-LAH_RxSWIN-429]

Für das Steuergerät mit zentralem Diagnosezugang gelten folgende zusätzliche Anforderungen:

2.3.4.1 RxSWIN

[allg. Anf.: F-LAH_RxSWIN-470]

Das Steuergerät mit dem zentralen Diagnosezugang muss den DataIdentifier "0xF18F-Regulation_x_software_identification_numbers" entsprechend Kapitel "Diagnoseobjekte" umsetzen.

2.3.4.2 Funktion "Diagnosefilter"

[I: F-LAH_RxSWIN-625]

Die Funktion Diagnosefilter ist im Gateway-Steuergerät umgesetzt und verhindert das Routing der Diagnoseanfragen mit Schreibanforderungen (Kodierung, Anpassung/Kalibrierung, Datensatzdownload, Programmierung) während der Fahrt. Der lesende Zugriff (z. B. Identifikation, Fehlerspeicher) sowie die Generic Scan Tool-Kommunikation ist ohne Einschränkung möglich.

[allg. Anf.: F-LAH_RxSWIN-476]

Basierend auf Dokument /9/ muss die Funktion Diagnosefilter im Gateway-Steuergerät mit folgenden Änderungen umgesetzt werden:

[allg. Anf.: F-LAH_RxSWIN-630]

- Das Deaktivieren des Diagnosefilters mittels Motorhauben-Kontaktschalter darf mit Umsetzung dieses Lastenheftes nicht mehr umgesetzt werden.

[allg. Anf.: F-LAH_RxSWIN-823]

- Das Deaktivieren des Diagnosefilters erfordert den SecurityLevel SFD-Zugriffschutz, Rolle BASIC.

[allg. Anf.: F-LAH_RxSWIN-767]

- Der aktivierte Diagnosefilter muss auch das Routing von Diagnoseanfragen mit Schreibanforderungen über serviceorientierter Diagnose verhindern.

[allg. Anf.: F-LAH_RxSWIN-824]

- Die DataIdentifier der Funktion Diagnosefilter sind entsprechend der Definition dieses Dokumentes zu unterstützen.

[allg. Anf.: F-LAH_RxSWIN-338]

- Das Deaktivieren des Diagnosefilters durch den Service SecurityAccess (27hex) ist nicht erlaubt.

[I: F-LAH_RxSWIN-541]

Hinweis: Es ist nur ein Zugriffsschutzverfahren in der Applikation eines Steuergerätes für die Diagnoseobjekte erlaubt. D. h. alle Diagnoseobjekte müssen einheitlich über den SFD-Zugriffsschutz geschützt werden. Gemäß Dokument /4/ erfordert die Umsetzung des SFD-Zugriffschutz auf dem Steuergerät mit dem zentralen Diagnosezugang, dass der Service SecurityAccess (Service 27hex) in der Applikation entfallen muss.

[allg. Anf.: F-LAH_RxSWIN-218]

Tabelle 2-2 Übersicht der Eigenschaften der Diagnoseobjekte für die Funktion "Diagnosefilter"

			DiagnosticSession		
			Default Session	NonDefaultSession	
				ECUProgram-mingSession	Extended-Diagnostic-Session
Diagnoseobjekte (DID)	SID	Security Level	0x01	0x02	0x03
0x0BEE-Diagnostic_filter_activation	22hex	NoImp	Available	notAvailable	Available
	2Ehex	SFD-Basic	notAvailable	notAvailable	Available
0x0BEF-Diagnostic_filter_vehicle_mileage_thresholds	22hex	NoImp	Available	notAvailable	Available
	2Ehex	at least SFD-Basic	notAvailable	notAvailable	Available
0x539B-Diagnostic_filter_status	22hex	NoImp	Available	notAvailable	Available
	2Ehex	NoImp	notAvailable	notAvailable	notAvailable
0x539C-Diagnostic_filter_life_cycle_data	22hex	NoImp	Available	notAvailable	Available
	2Ehex	NoImp	notAvailable	notAvailable	notAvailable
0x05EF-Diagnostic_filter_permanent_deactivation	22hex	NoImp	Available	notAvailable	Available
	2Ehex	SFD-Ext.	notAvailable	notAvailable	Available

2.3.4.2.1 0x0BEE-Diagnostic_filter_activation (Diagnosefilter, Aktivierung)

[I: F-LAH_RxSWIN-731]

Über diesen DataIdentifier wird die Funktion "Diagnosefilter" initial im Diagnose-Server aktiviert. Dieser DataIdentifier ist über den SecurityLevel "SFD-Zugriffsschutz, Rolle BASIC" geschützt.

[allg. Anf.: F-LAH_RxSWIN-696]

Tabelle 2-3 Aufbau DataRecord vom DataIdentifier 0x0BEE-Diagnostic_filter_activation

DID	0x0BEE
Bezeichnung	Diagnostic_filter_activation
Beschreibung	Dieser DataIdentifier dient zum Aktivieren/Deaktivieren der Funktion "Diagnosefilter".
Convention	Steuergerät mit zentralem Diagnosezugang
Diagnoseklasse	DK4-high
Session	APP: 0x01 (R), 0x03 (R/W) BLF: NoImp
SecurityLevel	SFD-Zugriffsschutz, Rolle BASIC
Changing	DIAG
Format	<[Diagnostic_filter_activation] 1-Byte-Hex>
Range	[Diagnostic_filter_activation] 00hex = Funktion ist nicht aktiv 01hex = Funktion ist aktiv 02hex - FFhex = VOLKSWAGEN AG-reserved
Init	00hex = Funktion ist nicht aktiv
Beispiel	01hex = Funktion ist aktiv
Datenkategorie	Werkstattparameter
eBZD	NoImp

2.3.4.2.2 0x0BEF-Diagnostic_filter_vehicle_mileage_thresholds (Diagnosefilter, Kilometerschwellen)

[I: F-LAH_RxSWIN-724]

Über diesen DataIdentifier wird die Kilometerschwelle konfiguriert, mit der der Diagnosefilter automatisch aktiviert wird. Dieser DataIdentifier ist mindestens über den SecurityLevel "SFD-Zugriffsschutz, Rolle BASIC" geschützt.

[allg. Anf.: F-LAH_RxSWIN-720]

Tabelle 2-4 Aufbau DataRecord vom DataIdentifier 0x0BEF-Diagnostic_filter_vehicle_mileage_thresholds

DID	**0x0BEF**
Bezeichnung	Diagnostic_filter_vehicle_mileage_thresholds
Beschreibung	Dieser DataIdentifier dient zum Konfigurieren der Kilometerschwelle für die automatische Aktivierung des Diagnosefilters.
Convention	Steuergerät mit zentralem Diagnosezugang
Diagnoseklasse	DK4-high
Session	APP: 0x01 (R), 0x03 (R/W) BLF: NoImp
SecurityLevel	SFD-Zugriffsschutz, mindestens Rolle BASIC
Changing	DIAG
Format	<<[Vehicle_mileage_production] 1-Byte-Hex>+ <[Vehicle_mileage_service] 1-Byte-Hex>>
Range	[Diagnostic_filter_production] 00hex - FFhex [Vehicle_mileage_service] 00hex - FFhex
Init	C8 14hex
Beispiel	00 14hex
Datenkategorie	Werkstattparameter
eBZD	NoImp

2.3.4.2.3 0x05EF-Diagnostic_filter_permanent_deactivation (Diagnosefilter, dauerhafte Deaktivierung)

[I: F-LAH_RxSWIN-216]

Über diesen DataIdentifier erfolgt das permanente Deaktivieren des Diagnosefilters. Dieser DataIdentifier ist über den SecurityLevel "SFD-Zugriffsschutz, Rolle EXTENDED" geschützt.

[allg. Anf.: F-LAH_RxSWIN-722]

Tabelle 2-5 Aufbau DataRecord vom DataIdentifier 0x05EF-Diagnostic_filter_permanent_deactivation

DID	**0x05EF**
Bezeichnung	Diagnostic_filter_permanent_deactivation
Beschreibung	Über diesen DataIdentifier wird der Diagnosefilter dauerhaft deaktiviert.
Convention	Steuergerät mit zentralem Diagnosezugang
Diagnoseklasse	DK4-high
Session	APP: 0x01 (R), 0x03 (R/W) BLF: NoImp
SecurityLevel	SFD-Zugriffsschutz, Rolle EXTENDED
Changing	DIAG
Format	<[Diagnostic_filter_permanent_deactivation] 1-Byte-Hex>
Range	[Diagnostic_filter_permanent_deactivation] 00hex = Filter not activated 01hex = Filter activated 02hex - FFhex = VOLKSWAGEN AG-reserved
Init	01hex
Beispiel	00hex = Filter nicht aktiviert
Datenkategorie	Werkstattparameter
eBZD	NoImp

2.3.4.2.4 0x539B-Diagnostic_filter_status (Diagnosefilter, Status)

[I: F-LAH_RxSWIN-665]

Das Deaktivieren des Diagnosefilters ist an verschiedene Bedingungen geknüpft. Über den DataIdentifier "0x539B-Diagnostic_filter_status" sind diese Bedingungen auszugeben. Zusätzlich sind über diesen DataIdentifier die Deaktivierungsbedingungen als Deaktivierungsgründe zu dokumentieren.

[allg. Anf.: F-LAH_RxSWIN-694]

Tabelle 2-6 Aufbau DataRecord vom DataIdentifier 0x539B-Diagnostic_filter_status

DID	**0x539B**
Bezeichnung	Diagnostic_filter_status
Beschreibung	Dieser DataIdentifier beinhaltet Informationen über den Status des Diagnosefilters. - Der Parameter [Funktionsstatus] gibt die Art der Funktionsaktivierung an. - Der Parameter [Filterstatus] gibt an, ob der DiagnoseFilter aktiv ist. - Der Parameter [Kilometerstand] enthält den aktuellen Wert des Kilometerzählers nach der Deaktivierung des Filters. - Der Parameter [Deaktivierungsgrund] enthält Angaben zum Deaktivierungsgrund des Filters.
Convention	Steuergerät mit zentralem Diagnosezugang
Diagnoseklasse	DK4-high
Session	APP: 0x01 (R), 0x03 (R) BLF: NoImp
SecurityLevel	NoImp
Changing	DIAG
Format	<<[Funktionsstatus] 1-Byte-Hex>+ <[Filterstatus] 1-Byte-Hex>+ <[Kilometerzähler] 1-Byte-Hex>+ <[Deaktivierungsgrund] 1-Byte-Hex>>
Range	[Funktionsstatus] 00hex = Funktion ist nicht aktiv 01hex = VOLKSWAGEN AG-reserved 02hex = Funktion aktiv - Filterdeaktivierung durch Motorhaube 03hex = Funktion aktiv - Filterdeaktivierung durch Tastenkombination 04hex = Funktion aktiv - Filterdeaktivierung durch Zugriffsschutzverfahren 05hex = Funktion aktiv - Filterdeaktivierung durch Motorhaube und Tastenkombination 06hex - FFhex= VOLKSWAGEN AG-reserved [Filterstatus] 00hex = Filter ist nicht aktiv 01hex = Filter ist aktiv 02hex - FFhex = VOLKSWAGEN AG-reserved [Kilometerzähler] 00hex-FFhex [Deaktivierungsgrund] 00hex = nicht deaktiviert 01hex = Motorhaube offen 02hex = Elektronische Zentralelektrik nicht vorhanden 03hex = Crash erkannt 04hex = Tastenkombination 05hex = Zugriffsschutzverfahren 06hex = Produktion 07hex = dauerhaft deaktiviert 08hex - FFhex = VOLKSWAGEN AG-reserved
Init	kein Init-Wert im Server
Beispiel	00 01 11 01 Funktionsstatus: Filterdeaktivierung durch Motorhaube Filterstatus: aktiv Kilometerstand: 17km Deaktivierungsgrund: offene Motorhaube
Datenkategorie	Analysedaten
eBZD	NoImp

2.3.4.2.5 0x539C-Diagnostic_filter_life_cycle_data (Diagnosefilter, Historiendaten)

[I: F-LAH_RxSWIN-723]

Über diesen DataIdentifier können Historiendaten ausgelesen werden.

[allg. Anf.: F-LAH_RxSWIN-721]

Tabelle 2-7 Aufbau DataRecord vom DataIdentifier 0x539C-Diagnostic_filter_life_cycle_data

DID	**0x539C**
Bezeichnung	Diagnostic_filter_life_cycle_data
Beschreibung	Dieser DataIdentifier beinhaltet Historiendaten des Diagnosefilter.
Convention	Steuergerät mit zentralem Diagnosezugang
Diagnoseklasse	DK4-high
Session	APP: 0x01 (R), 0x03 (R) BLF: NoImp
SecurityLevel	NoImp
Changing	DIAG
Format	<<[Life_cycle_data_1] 31-Byte-ASCII (80125)>+ <[Life_cycle_data_2] 31-Byte-ASCII (80125)>+ <[Life_cycle_data_3] 31-Byte-ASCII (80125)>+ <[Life_cycle_data_4] 31-Byte-ASCII (80125)>+ <[Life_cycle_data_5] 31-Byte-ASCII (80125)>>
Range	[Life_cycle_data_1] 31-Byte-ASCII (80125) [Life_cycle_data_2] 31-Byte-ASCII (80125) [Life_cycle_data_3] 31-Byte-ASCII (80125) [Life_cycle_data_4] 31-Byte-ASCII (80125) [Life_cycle_data_5] 31-Byte-ASCII (80125)
Init	2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2Dhex
Beispiel	-
Datenkategorie	Analysedaten
eBZD	NoImp

3 Anforderungen zur Absicherung der Fahrzeugdiagnose

3.1 SFD-E2E-Absicherung

[allg. Anf.: F-LAH_RxSWIN-82]

Diagnose-Server, die SFD-Ende-zu-Ende-Absicherung gemäß Kapitel "Anforderungen an Steuergeräte" umsetzen, müssen das Dokument /4/ (Q-LAH SFD Version 2.1 inkl. Errata zu Version 2.1) verwenden.

[allg. Anf.: F-LAH_RxSWIN-341]

Folgende Datenkategorien (Datenarten) gemäß Dokument /8/, die über den Service WriteDataByIdentifier (2Ehex) geschrieben werden, sind SFD-Ende-zu-Ende abzusichern und in den Diagnosedaten-Tabellen des BT-LAH entsprechend zu dokumentieren:

[allg. Anf.: F-LAH_RxSWIN-811]

- Codierung (ausstattungsabhängiges Ein/Ausschalten von Teilfunktionen)

[allg. Anf.: F-LAH_RxSWIN-810]

- Fahrzeugparameter (fahrzeugabhängige Parameter und Einstellung) die per Applikations-Datensätze übertragen werden

[allg. Anf.: F-LAH_RxSWIN-809]

- Erstbedatungswerte (einmalig (initial) beschreibbare Defaultwerte, Parameter und Einstellungen)

3.1.1 Sonderfall für DK4-low/DK4V-low-Systeme mit unterlagerten DK2/DK2F/DK2FV-Systemen

[I: F-LAH_RxSWIN-932]

Bei den folgenden Abbildungen handelt es sich um beispielhafte Teil-Abläufe, die nicht implementiert werden dürfen.

[I: F-LAH_RxSWIN-101]

DK2/DK2F/DK2FV-Systeme setzen den Schutz der Fahrzeugdiagnose (SFD) mit der Ende-zu-Ende Absicherung nicht selbst um.

[allg. Anf.: F-LAH_RxSWIN-100]

Ein DK4-low/DK4V-low-System muss die, über den Service WriteDataByIdentifier (2Ehex) schreibbaren, Konfigurationsdaten eines unterlagerten DK2/DK2F/DK2FV-Systems entsprechend der Anforderung "F-LAH_RxSWIN-341" vor nicht autorisiertem Zugriff mittels SFD-Ende-zu-Ende-Absicherung schützen. Abweichungen sind mit der Security-Abteilung des Auftraggebers abzustimmen.

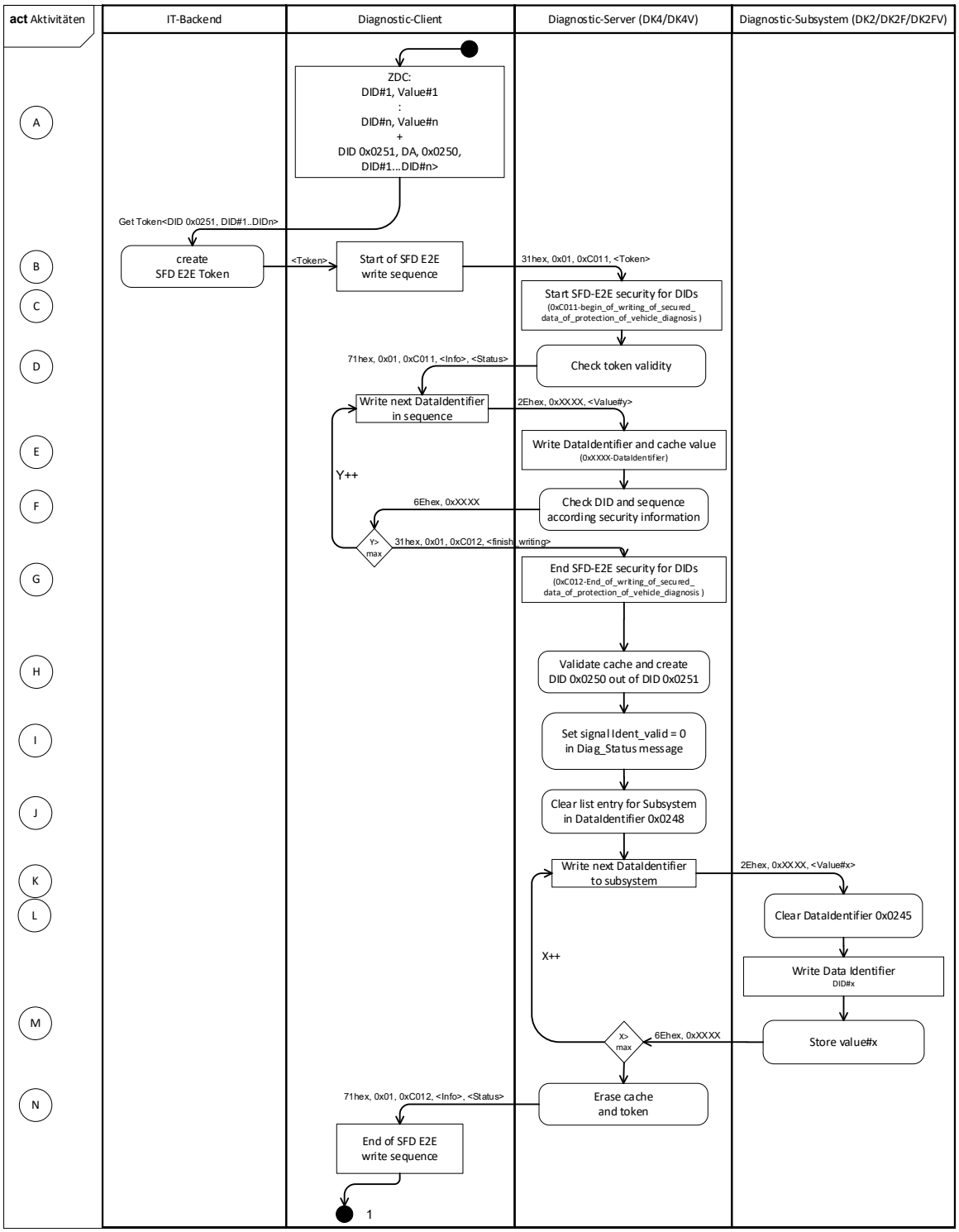
3.1.1.1 Schreiben von SFD-Ende-zu-Ende abgesicherten Daten in ein DK2/DK2F/DK2FV-System

[I: F-LAH_RxSWIN-280]

Ein Diagnose-Client schreibt SFD-Ende-zu-Ende abgesicherte Konfigurationsdaten über ein DK4-low/DK4V-low-System in ein unterlagertes DK2/DK2F/DK2FV-System.

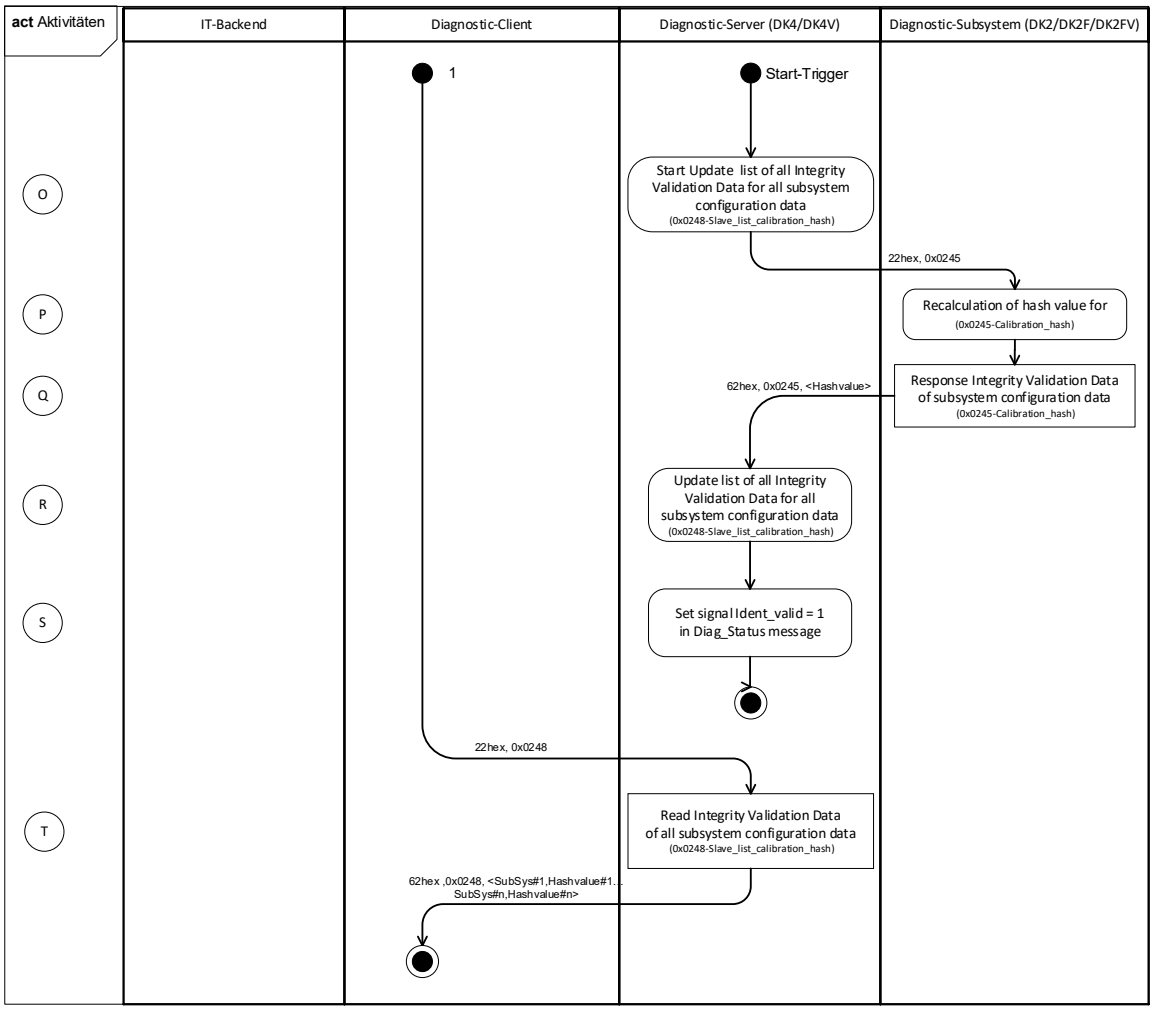
[I: F-LAH_RxSWIN-281]

Abbildung 3-1 Schreiben von Anpassungen in ein DK2/DK2F/DK2FV-System, Teil 1/2



[I: F-LAH_RxSWIN-282]

Abbildung 3-2 Schreiben von Anpassungen in ein DK2/DK2F/DK2FV-System, Teil 2/2



[I: F-LAH_RxSWIN-283]

A – Die für die Steuergeräte-Konfiguration notwendigen DataIdentifier werden dem Diagnose-Client zur Verfügung gestellt. Die DataIdentifier ergeben sich durch den Bauauftrag, der den ZDC konfiguriert. Hieraus resultiert auch der Inhalt des DataIdentifier "0x0250-Integrity_validation_data_configuration_list" für ein bestimmtes Subsystem. Der Diagnostic-Client fragt ein Absicherungs-Token am SFD-IT-Backend für den Diagnostic-Server an.

[I: F-LAH_RxSWIN-284]

B - Das IT-Backend erstellt ein Absicherungs-Token über die DataIdentifier DID#1 ... DID#n und DID "0x0251-Write_generic_to_sub_system".

[I: F-LAH_RxSWIN-285]

C* - Der Absicherungs-Token wird mit dem Start der SFD-Routine "0xC011-Begin_of_writing_of_secured_data_of_protection_of_vehicle_diagnosis" an den Diagnostic-Server übertragen und leitet die Schreib-Sequenz, für die SFD-Ende-zu-Ende abgesicherte Daten, ein.

[I: F-LAH_RxSWIN-286]

D* - Der Diagnostic-Server prüft die Validität des empfangenen SFD Absicherungs-Tokens. Entsprechend der SFD-E2E Absicherung gemäß Dokument /4/ wird bei einem Empfang eines ungültigen Tokens die Absicherungsinformationen gelöscht.

[I: F-LAH_RxSWIN-287]

E - Alle DataIdentifier, aus dem konfigurierten ZDC und der DataIdentifier "0x0251-Write_generic_to_sub_system" werden sequenziell mittels Service WriteDataByIdentifier (2Ehex) an den Diagnostic-Server übertragen und zwischengespeichert. Bei SFD mit Gruppenfreischaltung werden sie direkt vom Diagnostic-Server in die Diagnostic-Subsysteme geschrieben.

[I: F-LAH_RxSWIN-288]

F* - Entsprechend der SFD-E2E Absicherung nach Dokument /4/ wird die Empfangsreihenfolge der einzelnen DataIdentifier entsprechend der Absicherungsinformation überprüft. Bei positiver Prüfung antwortet der Diagnostic Server mit einer positiven Response. Bei einem negativen Prüfergebnis werden die empfangen DataIdentifier und die zugehörigen DataRecords sowie die Absicherungsinformationen gelöscht. Bei einem negativen Prüfergebnis antwortet der Diagnostic-Server mit einem NRC 0x22 (ConditionsNotCorrect).

[I: F-LAH_RxSWIN-289]

G* - Die Schreibsequenz des Diagnostic-Clients wird mit dem Start der Routine "0xC012-End_of_writing_of_secured_data_of_protection_of_vehicle_diagnosis" mit der RoutineControlOption [0x01-Finish_writing_of_secured_data] abgeschlossen.

[I: F-LAH_RxSWIN-290]

H* - Die Authentizität der übertragenen DataIdentifiers wird anhand der Absicherungsinformation aus dem Absicherungs-Token geprüft. Die Adressierung des Ziel-Subsystems und der Inhalt des DataIdentifier "0x0250-Integrity_validation_data_configuration_list" wird anhand des empfangenen DataIdentifier "0x0251-Write_generic_to_sub_system" ermittelt.

[I: F-LAH_RxSWIN-295]

I - In der Diag_Status-Botschaft des Diagnostic-Server wird im Signal [Ident_valid] mit dem Wert '0' angezeigt, dass die Identifikationsdaten aktuell nicht gültig sind. Durch das Speichern der empfangenen DataIdentifier ist der vorhanden Hashwert der Konfigurationsdaten für das Subsystem nicht mehr gültig und muss neu berechnet werden.

[I: F-LAH_RxSWIN-716]

J - Der Listen-Eintrag im Sammel-DataIdentifier "0x0248-Slave_list_configuration_hash" für das Diagnostic Subsystem wird gelöscht.

[I: F-LAH_RxSWIN-293]

K- Die zwischengespeicherten DataIdentifier aus dem Zwischenspeicher werden sequentiell zu dem Diagnostic-Subsystems mittels Service WriteDataByIdentifier (2Ehex) gesendet.

[I: F-LAH_RxSWIN-717]

L - Mit dem Request zum Schreiben der Konfigurationsdaten wird der DataIdentifier "0x0245-Configuration_hash" im Diagnostic Subsystem wird gelöscht.

[I: F-LAH_RxSWIN-291]

M - Das Diagnostic-Subsystem prüft die im Request empfangenen Daten auf Gültigkeit (z.B. Wertebereich). Bei einer positiven Prüfung werden die Daten im Subsystem gespeichert. Bei einer negativen Prüfung werden die Daten verworfen und ein negative Response zu dem Diagnostic-Server gesendet. Der Diagnostic-Server sendet einen Fehlercode im positiven Response der Routine "0xC012-End_of_writing_of_secured_data_of_protection_of_vehicle_diagnosis"zu dem Diagnostic-Client.

[I: F-LAH_RxSWIN-296]

N - Der Zwischenspeicher und die Absicherungsinformationen in dem Diagnostic-Server werden gelöscht, nachdem alle empfangenen DataIdentifier in den Zielspeicher des Diagnostic-Subsystem übertragen wurden.

[I: F-LAH_RxSWIN-297]

O - Das Beenden der SFD-Schreibsequenz ist der "Start-Trigger" des Diagnostic-Server um den Sammel-DataIdentifier „0x0248-Slave_list_configuration_hash“ zu aktualisieren, indem er den DataIdentifier "0x0245-Configuration_hash" des Diagnostic-Subsystems ausliest. ("Start-Trigger" siehe Anforderungen F-LAH_RxSWIN-200 und F-LAH_RxSWIN-693)

[I: F-LAH_RxSWIN-298]

P - Die Abfrage des DataIdentifier "0x0245-Configuration_hash" dient als Trigger zur Neuberechnung des Hashwertes der Konfigurationsdaten. Für die Neuberechnung werden genau die DataIdentifier, in genau der Reihenfolge, aus der Liste, die mit "0x0250-Integrity_validation_data_configuration_list" übertragen wurden, einbezogen.

[I: F-LAH_RxSWIN-299]

Q - Solange die Neuberechnung des Hashwertes noch nicht abgeschlossen ist, antwortet das Diagnostic-Subsystem mit einem NRC 0x78 (Response Pending) dem Busmaster. Wenn die Neuberechnung abgeschlossen ist wird der Request mit einem positiven Response beantwortet.

[I: F-LAH_RxSWIN-300]

R - Der Listeneintrag für das Diagnostic-Subsystem in dem Sammel-DataIdentifier "0x0248 Slave_list_configuration_hash" im Diagnostic-Server wird anhand des DataIdentifier "0x0245-Configuration_hash" aktualisiert.

[I: F-LAH_RxSWIN-301]

S - In der Diag_Status Botschaft des Diagnostic-Server wird im Signal [Ident_valid] mit dem Wert "1" angezeigt, dass die Identifikationsdaten vollständig gültig sind.

[I: F-LAH_RxSWIN-695]

T - Der Diagnose-Client fragt den Sammel-DataIdentifier "0x0248-Slave_list_configuration_hash" für alle Diagnostic-Subsysteme ab.

[I: F-LAH_RxSWIN-304]

*) Nur relevant für SFD-E2E abgesicherte Diagnose-Systeme. Nicht relevant für Diagnose-Systeme mit Gruppenfreischaltung in der Produktion, vor ZP8.

3.2 ECU programming data security

[allg. Anf.: F-LAH_RxSWIN-860]

Für neue Plattformen gilt:

[allg. Anf.: F-LAH_RxSWIN-912]

- Diagnose-Server müssen das Dokument /5/ (Q-LAH FDS) mindestens in der Version 3.1 oder ein alternatives, mindestens gleichwertiges Absicherungsverfahren, welches mit der E/E-Security-Abteilung des Auftraggebers abgestimmt ist, verwenden.

[allg. Anf.: F-LAH_RxSWIN-862]

- Diagnose-Server müssen das Dokument /6/ (Q-LAH 80126) mindestens in der Version 2.6 verwenden.

[allg. Anf.: F-LAH_RxSWIN-864]

- Diagnose-Server müssen das Dokument /14/ (Q-LAH 80128-3) mindestens in der Version 4.2 verwenden.

[allg. Anf.: F-LAH_RxSWIN-913]

Für Bestandsplattformen gilt:

[allg. Anf.: F-LAH_RxSWIN-914]

- Diagnose-Server müssen das Dokument /5/ (Q-LAH FDS) mindestens in der Version 2.1 oder ein alternatives, mindestens gleichwertiges Absicherungsverfahren, welches mit der E/E-Security-Abteilung des Auftraggebers abgestimmt ist, verwenden.

[allg. Anf.: F-LAH_RxSWIN-915]

- Diagnose-Server müssen das Dokument /6/ (Q-LAH 80126) mindestens in der Version 2.3 verwenden.

[allg. Anf.: F-LAH_RxSWIN-916]

- Diagnose-Server müssen das Dokument /14/ (Q-LAH 80128-3) mindestens in der Version 4.0 verwenden.

4 Integrity Validation Data

[I: F-LAH_RxSWIN-786]

Integrity Validation Data werden verwendet:

[I: F-LAH_RxSWIN-831]

- als Integritätsmerkmal von Software und Daten

[I: F-LAH_RxSWIN-830]

- zur Überprüfung der Integrität von Steuergeräten in der Produktion

[I: F-LAH_RxSWIN-829]

- zur Identifikation der Konfiguration von Steuergeräten per Zieldaten-Container

4.1 Allgemeine Anforderungen

[I: F-LAH_RxSWIN-225]

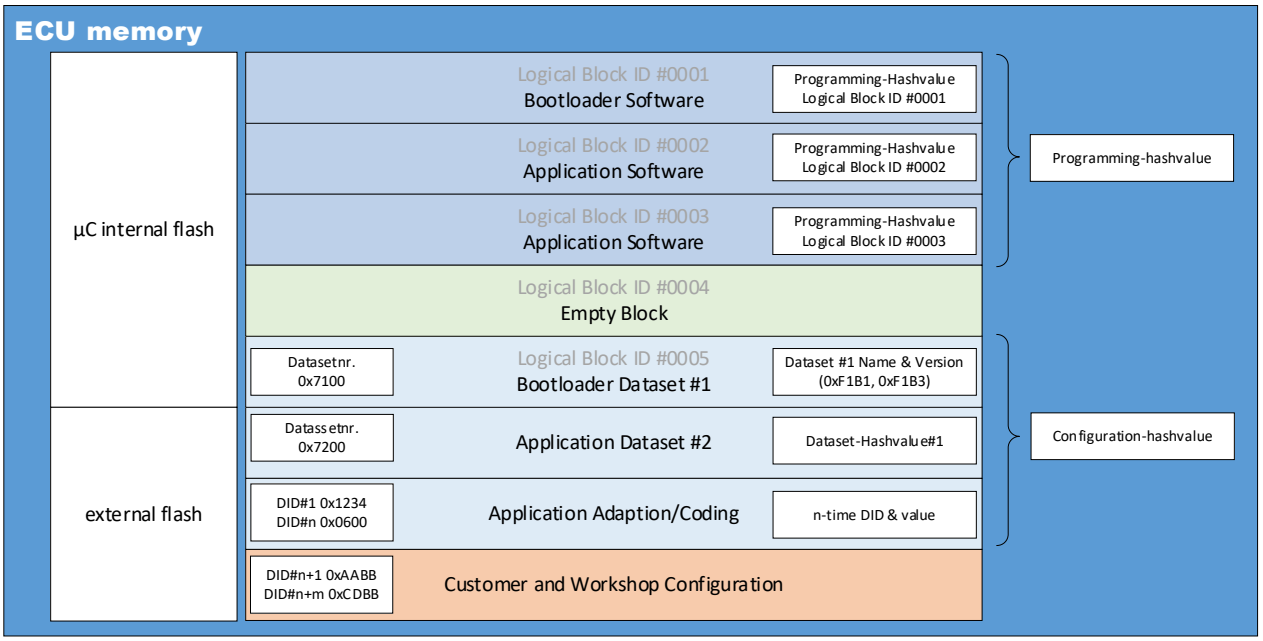
Für Instruction-Code (Applikations-und Bootloader-Software) und Konfigurationsdaten (DataIdentifier und Datensätze aus dem ZDC) werden separate Hashwerte berechnet. Die Berechnung der Hashwerte erfolgt sowohl in IT-Systemen und im Diagnose-Server. Dadurch ist ein Abgleich zwischen dem Sollzustand aus dem IT-Backend und dem Istzustand in den Diagnose-Servern möglich.

[I: F-LAH_RxSWIN-233]

Jeder der Hashwerte wird vom Diagnose-Server als Integrity Validation Data zur Verfügung gestellt. Dies erfolgt je nach Diagnoseklasse per DataIdentifier oder RoutineIdentifier.

[I: F-LAH_RxSWIN-234]

Abbildung 4-1 Beispielhafte Steuergeräte-Speicherarchitektur



[allg. Anf.: F-LAH_RxSWIN-638]

Zur Berechnung des Programmierungs- und des Konfigurations-Hashwertes wird der SHA-256 Algorithmus verwendet.

[allg. Anf.: F-LAH_RxSWIN-707]

Die Berechnung des Hashwertes darf maximal 100 ms betragen.
Hinweis: Kann das Steuergerät aufgrund der Berechnung des Hashwertes nicht innerhalb von 50ms antworten, ist mit einem negativen Response Code "0x78-RequestCorrectlyReceived-Response-Pending" zu antworten. Abweichungen von dieser Dauer müssen mit der Diagnosefachabteilung sowie mit Produktion und Kundendienst abgestimmt werden und in den Komponentenspezifischen Diagnoseanforderungen des BT-LAH dokumentiert werden.

[I: F-LAH_RxSWIN-940]

Tabelle 4-1 Zuordnung von Datenkategorie (Datenart nach Dokument /8/) zu Hashwert

	Programmierungs-Hashwert	Konfigurations-Hashwert
Programmdaten	x	NoImp
Applikationsdaten	x	NoImp
Codierung	NoImp	x
Fahrzeugparameter	NoImp	x
Erstbedatungswerte	NoImp	x
Kundenparameter	NoImp	NoImp
Werkstattparameter	NoImp	NoImp
Prozessparameter	NoImp	NoImp
Lernwerte	NoImp	NoImp
Analysedaten	NoImp	NoImp

4.1.1 Programmierungs-Hashwert

4.1.1.1 Anforderungen an Embedded-Systeme

[allg. Anf.: F-LAH_RxSWIN-423]

Die Anforderungen dieses Kapitel gelten für alle programmierbaren Diagnose-Server entsprechend Dokument /6/.

[allg. Anf.: F-LAH_RxSWIN-238]

Integrity Validation Data für die Programmierung (Instruction-Code) umfasst grundsätzlich alle logischen Blöcke der Applikations- und Bootloadersoftware.

[allg. Anf.: F-LAH_RxSWIN-232]

Für alle logischen Blöcke der Applikations- und Bootloadersoftware im Speicher eines Steuergerätes entsprechend Dokument /6/ wird ein einziger Hashwert durch das Steuergerät berechnet.

[allg. Anf.: F-LAH_RxSWIN-740]

Bei Verwendung von segmentierten Blöcken entsprechend Dokument /6/ erfolgt die Berechnung der Integrity Validation Data immer über den gesamten logischen Block inklusive aller Segmente.

[allg. Anf.: F-LAH_RxSWIN-570]

Als Bestandteil der Applikations- bzw. Bootloader Software werden auch die Metadaten (zur SW-Konfiguration z. B. für Adaptive Autosar oder Switch-Konfigurationen) und die Default-Daten und Default-Datensätze gezählt.

[allg. Anf.: F-LAH_RxSWIN-784]

Bei einem 2-stufigen Bootloader-Update werden nur die logische Blöcke der Applikation zur Berechnung des Programmierungs-Hashwert aus dem zweitem Flash-pdx verwendet.

[allg. Anf.: F-LAH_RxSWIN-389]

Leere logische Blöcke, die als Vorhalt dienen, dürfen nicht in die Hashwertberechnung einbezogen werden.

[allg. Anf.: F-LAH_RxSWIN-745]

Flash-Treiber, als logischer Block im RAM, dürfen nicht in die Hashwertberechnung einbezogen werden.

[allg. Anf.: F-LAH_RxSWIN-741]

Abweichend zu den Anforderungen [A: 80126-A915] und [A: 80126-A914] im Dokument /6/ dürfen optionale nicht leere Blöcke, z.B. für Zusatzschriften, die nicht direkt zur Applikation gehören und die Funktionsfähigkeit nicht beeinflussen, nicht von der Kompatibilitätsprüfung durch die Routine "0xFF01-Check Programming Dependencies" ausgenommen werden und müssen in der Hashwertberechnung berücksichtigt werden.

[allg. Anf.: F-LAH_RxSWIN-768]

Abweichend zu der Anforderung [A: 80126-A146] im Dokument /6/ muss der CRC32-Wert über die unkomprimierten und unverschlüsselten Nutzdaten in der Prüfanforderung bedatet sein.

[allg. Anf.: F-LAH_RxSWIN-769]

Hinweis: Sind innerhalb eines logischen Blocks Daten (z. B. lieferantenspezifische Informationen) enthalten, die nicht bei der CRC32-Berechnung beim Lieferanten berücksichtigt werden, dann dürfen diese Daten auch nicht bei der CRC32-Berechnung im Diagnose-Server berücksichtigt werden.

[allg. Anf.: F-LAH_RxSWIN-390]

In die Hashwertberechnung durch die Routine "0x0253-Calculate_integrity_validation_data", mit der ControlOption "Type_of_calculation" = 0x01, dürfen nur logische Blöcke einbezogen werden, die im DataIdentifier "0xF1AB-Logical Software Block Version" mit einer gültigen Software Versionsnummer gekennzeichnet sind.

[allg. Anf.: F-LAH_RxSWIN-692]

Feste, nicht durch die VOLKSWAGEN AG updatebare, Softwareanteile werden bei der Berechnung des Programmierungs-Hashwertes nicht berücksichtigt.

[allg. Anf.: F-LAH_RxSWIN-567]

Durch die Routine "0x0253-Calculate_integrity_validation_data", mit der ControlOption "Type_of_calculation" = 0x01, wird die Neuberechnung des Programmierungs-Hashwertes auf dem Steuergerät durchgeführt.

[allg. Anf.: F-LAH_RxSWIN-564]

Mit dem Starten der Routine "0xFF00-Erase Memory", die einen logischen Block mit Instruction-Code adressiert, muss die CRC32-Checksumme des adressierten logischen Blocks gelöscht werden.

[allg. Anf.: F-LAH_RxSWIN-565]

Mit der positiven Prüfung, durch die Routine "0x0202-Check Memory", dass der logische Block fehlerfrei übertragen wurde muss die CRC32-Checksumme des logischen Blocks neu berechnet werden.

[allg. Anf.: F-LAH_RxSWIN-718]

Mit dem positiven Response für die Routine "0x0202-Check Memory", die einen logischen Block mit Instruction-Code adressiert, wird die berechnete CRC32-Checksumme des logischen Blocks persistent gespeichert.

[allg. Anf.: F-LAH_RxSWIN-636]

Die Routine "0x0544-Verify_partial_software_checksum" entsprechend Dokument /6/ ist umzusetzen.

[allg. Anf.: F-LAH_RxSWIN-620]

Abweichend zur Dokument /6/ und Dokument /2/ muss die Routine "0x0544-Verify_partial_software_checksum" in der "DefaultSession (0x01)" in der Applikation zur Verfügung stehen.

[allg. Anf.: F-LAH_RxSWIN-617]

Mit dem Empfang des Requests für die Routine "0x0544-Verify_partial_software_checksum" muss die Neuberechnung der CRC32-Checksumme für den im Request adressierten logischen Block durchgeführt und persistent im Steuergerät abgespeichert werden.

[allg. Anf.: F-LAH_RxSWIN-345]

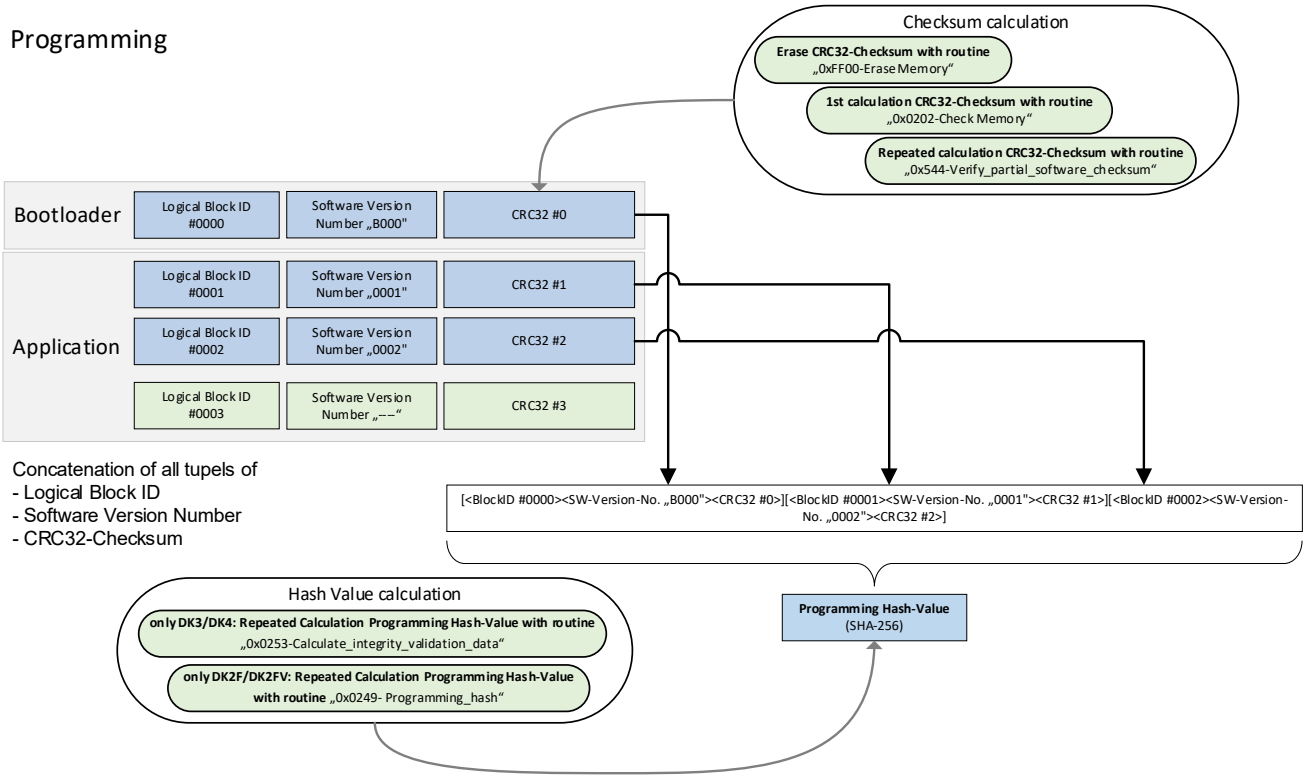
Die Berechnung des Programmierungs-Hashwertes erfolgt über die konkatenierten Tupel, bestehend aus der Block ID, Software Version Number und CRC32-Checksumme jedes logischen Blocks.

[allg. Anf.: F-LAH_RxSWIN-715]

Die Tupel zur Berechnung des Programmierungs-Hashwertes werden in aufsteigender Reihenfolge der logischen Block ID konkateniert.

[I: F-LAH_RxSWIN-235]

Abbildung 4-2 Konkatenation der Daten der logischen Blöcke der Applikation und des Bootloaders (Instruction-Code) zur Hashwertberechnung



[I: F-LAH_RxSWIN-388]

Hinweis:

- logische Block ID (2 Byte) gemäß Dokument /6/
- Programming-Hash-Value #n = Hashwert über den logischen Block der Applikations- bzw. der Bootloader-Software

[Prozess-Anf.: F-LAH_RxSWIN-770]

Die Berechnung des Soll-Programmierungs-Hashwertes erfolgt im IT-System analog der Berechnung im Diagnose-Server. Die CRC-32-Werte müssen dazu dem Flash-pdx entnommen werden.

[allg. Anf.: F-LAH_RxSWIN-659]

Die standardisierten Identifikationsdaten nach Dokument /2/ und Dokument /12/ sind bei der Berechnung des Programmierungs-Hashwertes unter den folgenden Bedingungen (UND-Verknüpfung) zu berücksichtigen, wenn sie:

[allg. Anf.: F-LAH_RxSWIN-817]

- weder über den Service WriteDataByIdentifier (2Ehex) noch über den Datensatzdownload (Bootloader/Applikation) änderbar sind

[allg. Anf.: F-LAH_RxSWIN-816]

- nur über Updateprogrammierung änderbar sind.

[allg. Anf.: F-LAH_RxSWIN-815]

- entsprechend der Diagnoseklasse gefordert sind.

[allg. Anf.: F-LAH_RxSWIN-814]

- entsprechend der OBD-Klasse gefordert sind.

[allg. Anf.: F-LAH_RxSWIN-813]

- entsprechend der Umsetzungsanforderungen für bestimmte Anwendungsfälle gefordert sind.

[allg. Anf.: F-LAH_RxSWIN-812]

- entsprechend der Systemauslegung gem. Dokument /12/ gefordert sind.

[I: F-LAH_RxSWIN-660]

Beispielhaft werden die folgenden DataIdentifier gem. Dokument /2/ und Dokument /12/ in Abhängigkeit der OBD-Relevanz, der Diagnoseklasse, der Umsetzungsanforderungen und der Systemauslegung dem Programmierungs-Hashwert zugeordnet.

Nur über Updateprogrammierung änderbar (Diagnoseklassen DK2F und höher):

- 0xF187 - VW Spare Part Number
- 0xF189 - VW Application Software Version Number
- 0xF19E - ASAM ODX File Identifier
- 0xF1A2 - ASAM ODX File Version

Nur für Systeme, die Software Compositions enthalten:

- 0x0441 - SWCO_list_nr
- 0x011D - SWCO_list_system_name

Nur für OBD-relevante Systeme:

- 0x02CE - OBD_type
- 0x02CF - OBD_class_description

[allg. Anf.: F-LAH_RxSWIN-688]

In die Hashwertberechnung der Programmdaten werden folgende Datenkategorien (Datenarten) gemäß Dokument /8/ einbezogen:

[allg. Anf.: F-LAH_RxSWIN-938]

- Programmdaten (Programmcode oder Software für das Steuergerät)
  Hinweis: Programmdaten sind Bestandteil des Flashcontainers.

[allg. Anf.: F-LAH_RxSWIN-939]

- Applikationsdaten (ausstattungsabhängige Daten und Kennlinien)
  Hinweis: Applikationsdaten sind Bestandteil des Flashcontainers.

4.1.1.2 Anforderungen an nonEmbedded-Systeme bzw. filebasierte Systeme

4.1.1.2.1 Verwendung von Flashcontainer "ODX-Flash/PDX-Flash"

[allg. Anf.: F-LAH_RxSWIN-641]

Bei Verwendung von Flashcontainer nach Dokument /14/ ist das Verfahren zur Berechnung des Programmierungs-Hashwertes entsprechend Kapitel "Anforderungen an Embedded-Systeme" anzuwenden.

4.1.1.2.2 Verwendung von Flashcontainer abweichend von "ODX-Flash/PDX-Flash"

[allg. Anf.: F-LAH_RxSWIN-643]

Bei Verwendung von Flashcontainer abweichend von Dokument /14/ kann die Berechnung systemspezifisch erfolgen.

[allg. Anf.: F-LAH_RxSWIN-644]

Der Soll-Wert des Programmierungs-Hashwertes muss in den Beschreibungsdateien (Manifest oder Meta-Daten) für die IT-Systeme auswertbar, enthalten sein.

[allg. Anf.: F-LAH_RxSWIN-645]

Die Implementierungen der Hashwertberechnung ist mit dem Auftraggeber (Diagnosefachabteilung und Konzern-IT) abzustimmen.

[Prozess-Anf.: F-LAH_RxSWIN-773]

Bei Verwendung von Flashcontainern im lum-f Format soll ein Backend-System die Checksummendatei des Mainmanifests zur Berechnung des Programmierungs-Hashwertes verwenden. Hierzu soll das Backend-System den Programmierungs-Hashwert über die Datei „main.mnf.cks“ bilden.

[allg. Anf.: F-LAH_RxSWIN-774]

Der Diagnose-Server muss den kompletten Baum der Manifest-Dateien inklusive des Wurzelknotens („main.mnf“) sowie dessen Metadaten (main.mnf.cks, main.mnf.cks.sig) auf das zu flashende System übertragen und zusammen mit den Programmfiles ablegen.

[allg. Anf.: F-LAH_RxSWIN-775]

Der Diagnose-Server muss bei jedem Zugriff auf die Manifest-Dateien die Integrität dieser Daten sicherstellen. Hierzu muss das zu programmierende System die Signatur des Wurzelknotens prüfen (Soll-Wert ist in der Datei main.mnf.cks.sig hinterlegt).

[allg. Anf.: F-LAH_RxSWIN-776]

Der Diagnose-Server muss für jedes, in den Manifestdaten, referenzierte Paket (Modul) die CRC32-Checksumme regelmäßig zur Laufzeit überprüfen.

[allg. Anf.: F-LAH_RxSWIN-777]

Eine Prüfung zur Laufzeit ist nicht notwendig, wenn der Diagnose-Server für das betroffene Paket (Modul) eine Überprüfung durch ein SecureBoot-Verfahren (z.B. dm_verity) durchführt.

[allg. Anf.: F-LAH_RxSWIN-779]

Durch die Routine "0x0253-Calculate_integrity_validation_data", mit der RoutineControlOption "Type_of_calculation" = 0x01, wird die Neuberechnung des Programmierungs-Hashwertes auf dem Diagnose-Server durchgeführt. Der Diagnose-Server soll hierbei folgende Fälle unterscheiden:

[allg. Anf.: F-LAH_RxSWIN-780]

- Es existiert noch kein vollständiges Prüfergebnis zur Laufzeit der Werte. In diesem Fall muss der Diagnose-Server den Hashwert der Checksummendatei des Main-Manifest („main.mnf.cks“) zurückgeben.

[allg. Anf.: F-LAH_RxSWIN-781]

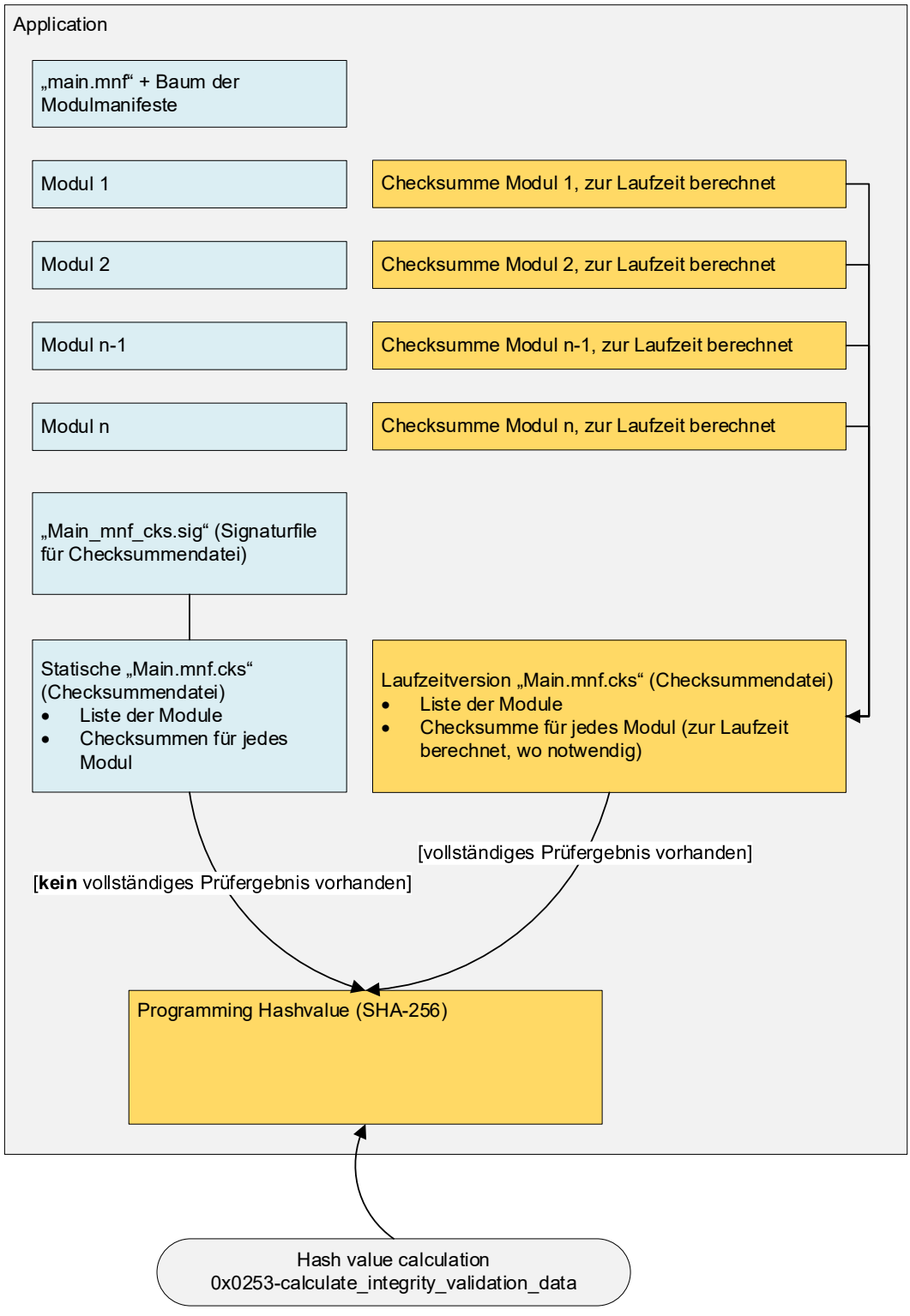
- Es existiert ein Prüfergebnis zur Laufzeit und der Diagnose-Server hat die Prüfung aller zu prüfenden Pakete (Module) positiv abgeschlossen: Der Diagnose-Server meldet das Ergebnis analog a).

[allg. Anf.: F-LAH_RxSWIN-782]

- Es existiert ein Prüfergebnis zur Laufzeit und der Diagnose-Server hat die Prüfung ein oder mehrerer zu prüfender Pakete (Module) negativ abgeschlossen: Der Diagnose-Server meldet einen Programmierungs-Hashwert zurück, der nicht mit der Sollvorgabe im IT-Backend übereinstimmt.

[I: F-LAH_RxSWIN-924]

Abbildung 4-3 Hashwertberechnung für lum-f container



4.1.2 Konfigurations-Hashwert

[allg. Anf.: F-LAH_RxSWIN-424]

Die Anforderungen dieses Kapitel gelten für alle konfigurierbaren Diagnoseserver.

[I: F-LAH_RxSWIN-646]

Es wird nicht zwischen embedded- und filebasierten Systemen unterschieden.

[allg. Anf.: F-LAH_RxSWIN-239]

Der Hashwert für Konfigurationsdaten umfasst logische Blöcke mit Bootloader-Datensätzen und Konfigurationsdaten der Applikation.
Hinweis: DK2F/DK2FV-Systeme besitzen keine Applikations- oder Bootloader-Datensätze.

[allg. Anf.: F-LAH_RxSWIN-241]

In die Hashwertberechnung der Konfigurationsdaten dürfen nur die folgenden Datenkategorien (Datenarten) gemäß Dokument /8/ einbezogen werden:

[allg. Anf.: F-LAH_RxSWIN-791]

- Codierung (ausstattungsabhängiges Ein/Ausschalten von Teilfunktionen)

  [allg. Anf.: F-LAH_RxSWIN-790]
- Fahrzeugparameter (fahrzeugabhängige Parameter und Einstellung)

  [allg. Anf.: F-LAH_RxSWIN-789]
- Erstbedatungswerte (einmalig (initial) beschreibbare Defaultwerte, Parameter und Einstellungen)

  [allg. Anf.: F-LAH_RxSWIN-308]

Nicht in die Hashwertberechnung der Applikations-Konfigurationsdaten dürfen die folgenden Arten von Daten gemäß Dokument /8/ einbezogen werden:

[allg. Anf.: F-LAH_RxSWIN-937]

- Programmdaten (Programmcode oder Software für das Steuergerät)
  Hinweis: Programmdaten sind Bestandteil des Programmierungs-Hashwert

  [allg. Anf.: F-LAH_RxSWIN-797]
- Applikationsdaten (ausstattungsabhängige Daten und Kennlinien)
  Hinweis: Programmdaten sind Bestandteil des Programmierungs-Hashwert

  [allg. Anf.: F-LAH_RxSWIN-796]
- Kundenparameter (kundenabhängige Parameter und Einstellungen)

  [allg. Anf.: F-LAH_RxSWIN-795]
- Werkstattparameter (werkstattabhängige Parameter und Einstellungen)
  Hinweis: Werkstattparameter dürfen nur innerhalb eines vorgegebenen Rahmens, ohne Beeinflussung der Typgenehmigung, verändert werden.

  [allg. Anf.: F-LAH_RxSWIN-794]
- Prozessparameter (prozessabhängige Parameter und Einstellungen)
  Hinweis: Prozessparameter mit Einfluss auf die Typgenehmigung müssen in der Bauzustandsdokumentation dokumentiert werden.

  [allg. Anf.: F-LAH_RxSWIN-793]
- Lernwerte (vom Steuergerät selbständig angelernte Parameter und Einstellungen)

  [allg. Anf.: F-LAH_RxSWIN-792]
- Analysedaten (während des normalen Fahrzeugbetriebs dynamisch erzeugte Daten)

[allg. Anf.: F-LAH_RxSWIN-785]

Nicht in die Hashwertberechnung der Applikations-Konfigurationsdaten darf der folgende DataIdentifier einbezogen werden:

- 0xF18F-Regulation_x_software_identification_numbers

[Prozess-Anf.: F-LAH_RxSWIN-787]

Hinweis: Der DataIdentifier 0xF18F ist nicht Bestandteil des ZDC vom Gateway-Steuergerät und kann somit nicht bei der Berechnung des Soll-Wertes für den Konfigurations-Hashwert berücksichtigt werden. Er wird über einen separaten Datencontainer zugesteuert und vollständig in der Bauzustandsdokumentation (BZD) erfasst.

[allg. Anf.: F-LAH_RxSWIN-544]

Eine Änderung von Kunden- und Werkstattparametern darf NICHT zu einer Änderung der Integrity Validation Data von Konfigurationsdaten führen.

[allg. Anf.: F-LAH_RxSWIN-568]

"Vorgegebene Datensätze" gemäß Dokument /7/ gehen in die Hashwertberechnung der Konfigurationsdaten mit ein und müssen im ZDC referenziert sein.

[allg. Anf.: F-LAH_RxSWIN-569]

"Vorgehaltene Datensätze" gemäß Dokument /7/ gehen in die Hashwertberechnung der Konfigurationsdaten mit ein und müssen im ZDC referenziert sein.
Hinweis: Für vorgehaltene Datensätze gilt die Forderung nach gültigen Software Versionsnummer im DataIdentifier "0xF1AB-Logical Software Block Version" nicht.

[allg. Anf.: F-LAH_RxSWIN-571]

"Default Datensätze" gemäß Dokument /7/ zählen zur Applikations-Software und dürfen nicht in die Hashwertberechnung der Konfigurationsdaten einbezogen werden.

[allg. Anf.: F-LAH_RxSWIN-772]

Reservierte Codierungen/Anpassungen gehen in die Hashwertberechnung der Konfigurationsdaten mit ein und müssen im ZDC enthalten sein.

[allg. Anf.: F-LAH_RxSWIN-691]

Bei einer Mehrfachablage von z. B. ASIL-relevanten Daten wird für die Berechnung des Konfigurations-Hashwertes der redundant abgelegte Teil der Daten nicht berücksichtigt.

[allg. Anf.: F-LAH_RxSWIN-647]

Die standardisierten Identifikationsdaten nach Dokument /2/ und Dokument /12/ sind bei der Berechnung des Konfigurations-Hashwertes unter den folgenden Bedingungen (UND-Verknüpfung) zu berücksichtigen, wenn sie:

[allg. Anf.: F-LAH_RxSWIN-822]

- über den Service WriteDataByIdentifier (2Ehex) oder über den Datensatzdownload (Bootloader/Applikation) änderbar sind.

[allg. Anf.: F-LAH_RxSWIN-821]

- entsprechend der Diagnoseklasse gefordert sind.

[allg. Anf.: F-LAH_RxSWIN-820]

- entsprechend der OBD-Klasse gefordert sind.

[allg. Anf.: F-LAH_RxSWIN-819]

- entsprechend der Umsetzungsanforderungen für bestimmte Anwendungsfälle gefordert sind.

[allg. Anf.: F-LAH_RxSWIN-818]

- entsprechend der Systemauslegung gem. Dokument /12/ gefordert sind.

[I: F-LAH_RxSWIN-651]

Beispielhaft werden die folgenden DataIdentifier nach Dokument /2/ und Dokument /12/ in Abhängigkeit der Diagnoseklasse, der Umsetzungsanforderungen und der Systemauslegung dem Konfigurations-Hashwert zugeordnet.

Nur für Busmaster-Systeme der Diagnoseklasse 4-high:
- 0x04A3 - Gateway Component List
- 0x061A - Slave_component_list

Nur für Systeme, die den Datensatzdownload der Generation 2 unterstützen:
- 0xF1B1 - VW_Application_data_set_identification
- 0xF1B3 - VW_Data_set_name

Nur für Systeme die Software Compositions enthalten:
- 0x0442 - SWCO_list

[allg. Anf.: F-LAH_RxSWIN-572]

Alle Konfigurationsdaten, die in die Hashwertberechnung der Konfigurationsdaten einbezogen werden, müssen Bestandteil des Zieldatencontainers sein. Weitere Konfigurationsdaten dürfen nicht im Zieldatencontainer enthalten sein.

[I: F-LAH_RxSWIN-599]

Hinweis: Im ZDC müssen alle Konfigurationsdaten enthalten sein, damit im Feld durch den Kundendienst die Möglichkeit gegeben ist, alle Konfigurationsdaten entsprechend der Typgenehmigung wiederherstellen zu können, falls der Hashwert im Steuergerät nicht mit dem Soll- Hashwert im IT-Backend übereinstimmt.

[allg. Anf.: F-LAH_RxSWIN-623]

Daten der Wegfahrsperre (WFS), des Komponentenschutzes (KS), des Vehicle Key Management Systems (VKMS) und Daten von Software als Produkt (SWaP) bzw. Function on Demand (FOD) dürfen nicht in die Hashwertberechnung der Konfigurationsdaten einbezogen werden.

[I: F-LAH_RxSWIN-624]

Hinweis: Die Daten der Wegfahrsperre (WFS) entsprechend Dokument /18/, des Komponentenschutzes (KS) entsprechend Dokument /19/, des Vehicle Key Management Systems (VKMS) entsprechend Dokument /15/ und Daten von Software als Produkt (SWaP) entsprechend Dokument /16/ bzw. Function on Demand (FOD) entsprechend Dokument /17/ werden abgesichert in das Fahrzeug eingebracht und in geschützten Speicherbereichen (SHE bzw. HSM) entsprechend Dokument /20/ abgelegt, so dass eine Manipulation der Daten ausgeschlossen werden kann.

[allg. Anf.: F-LAH_RxSWIN-346]

Über den DataIdentifier "0x0250-Integrity_validation_data_configuration_list" wird die Liste mit DataIdentifiern für Anpassung/Codierung sowie Datensatznummern vorgegeben, die für die Hashwertberechnung über die Konfigurationsdaten verwendet werden soll.

[allg. Anf.: F-LAH_RxSWIN-377]

Die Reihenfolge der Daten für die Hashwertberechnung ergibt sich aus dem Inhalt des DataIdentifiers "0x0250-Integrity_validation_data_configuration_list" und muss unverändert übernommen werden.

4.1.2.1 Berechnung der Einzel-Hashwerte bzw. Checksummen der Konfigurationsdaten

[I: F-LAH_RxSWIN-690]

Der Hashwert der Konfigurationsdaten wird mit Hilfe von verschiedenen einzelnen Hashwerten ermittelt. Diese Einzel-Hashwerte können über die RoutineIdentifier "0x0254-Calculate_individual_hash_value" ausgelesen werden.

4.1.2.1.1 Einzel-Hashwert für Anpassungen/Codierungen und Applikations-Datensätze gemäß Dokument /7/

[allg. Anf.: F-LAH_RxSWIN-576]

Die Einzel-Hashwerte für alle Anpassungen/Codierungen und Applikations-Datensätze der Konfigurationsdaten werden mit der positiven Response für die SFD-Routine "0xC012-End_of_writing_secured_data_of_protection_of_vehicle_diagnosis" erstmals berechnet.

[allg. Anf.: F-LAH_RxSWIN-577]

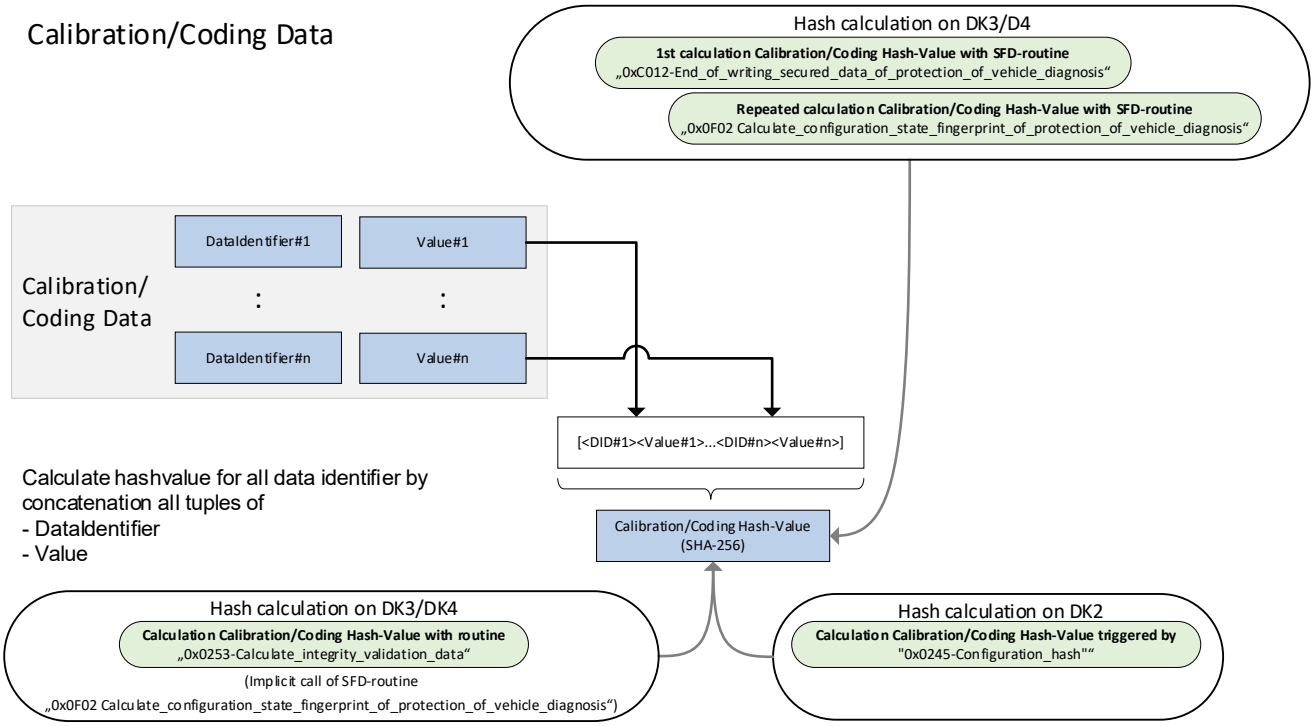
Die Einzel-Hashwerte für alle Anpassungen/Codierungen und Applikations-Datensätze der Konfigurationsdaten werden mit dem Empfang des Requests für die SFD-Routine "0x0F02-Calculate_configuration_state_fingerprint_of_protection_of_vehicle_diagnosis" erneut berechnet.

[allg. Anf.: F-LAH_RxSWIN-593]

Steuergeräte, die SFD-E2E nicht umgesetzt haben, berechnen die Einzel-Hashwerte für Anpassung/Codierung und Applikations-Datensätze, gemäß Dokument /7/ mit dem Empfang des Requests für die Routine "0x0253-Calculate_integrity_validation_data" mit der ControlOption "Type_of_calculation" = 0x00.

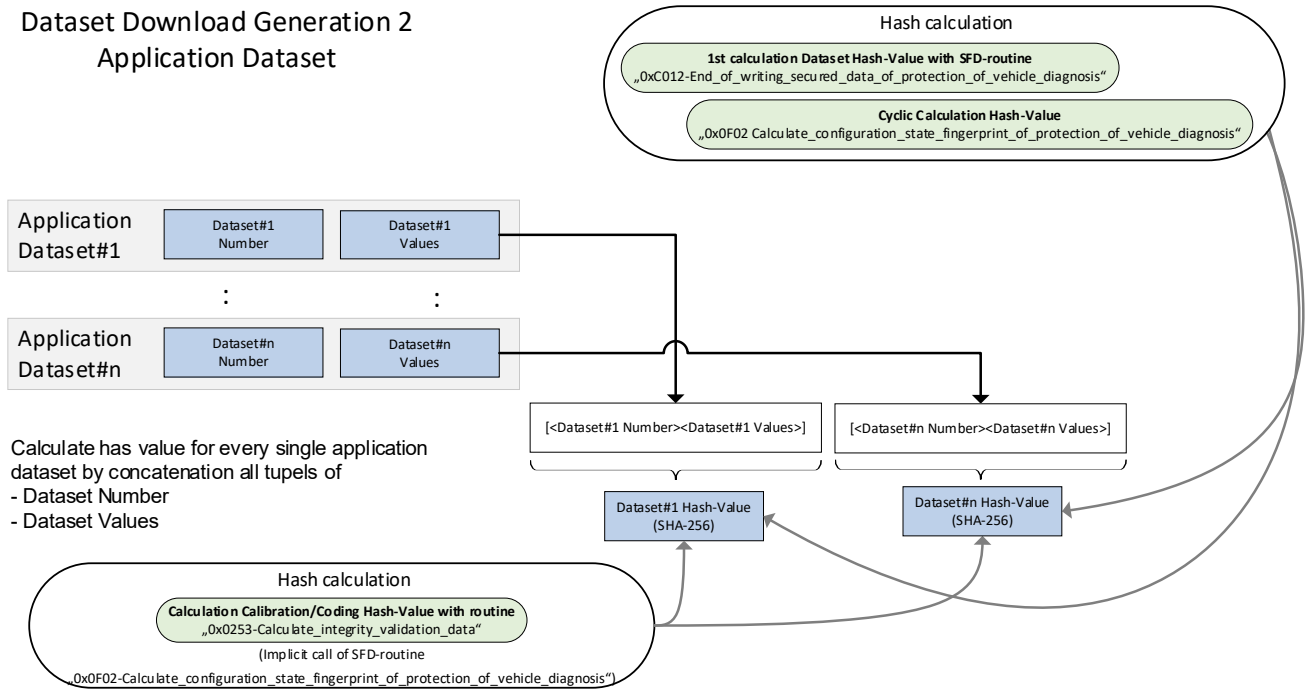
[I: F-LAH_RxSWIN-578]

Abbildung 4-4 Einzel-Hashwertberechnung über alle Anpassungen und Codierungen



[I: F-LAH_RxSWIN-585]

Abbildung 4-5 Einzel-Hashwertberechnung über Applikations-Datensätze



4.1.2.1.2 CRC-Checksumme für Datensätze der Datensatzdownload Generation 1 gemäß Dokument /11/

[allg. Anf.: F-LAH_RxSWIN-742]

Die im Datensatz enthaltene CRC16 bzw. CRC32-Checksumme darf nicht verändert werden.

[allg. Anf.: F-LAH_RxSWIN-743]

Im Steuergerät wird für jeden gültigen Datensatz zusätzliche eine CRC32-Checksumme über den gesamten Inhalt des Datensatzcontainer inklusive Daten, Checksumme und Speicheradresse berechnet und gespeichert.

[allg. Anf.: F-LAH_RxSWIN-580]

CRC32-Checksumme für einen Datensatz wird mit dem Empfang des Requests für die Routine "0x0300-Erase VW Memory" gelöscht.

[allg. Anf.: F-LAH_RxSWIN-581]

CRC32-Checksumme für einen Datensatz wird mit dem Empfang des Requests für die Routine "0x02FF-Calculate checksum" erstmals berechnet.

[allg. Anf.: F-LAH_RxSWIN-584]

CRC32-Checksumme für einen Datensatz wird mit der positiven Response für die Routine "0x02FF-Calculate checksum" persistent gespeichert.

[allg. Anf.: F-LAH_RxSWIN-582]

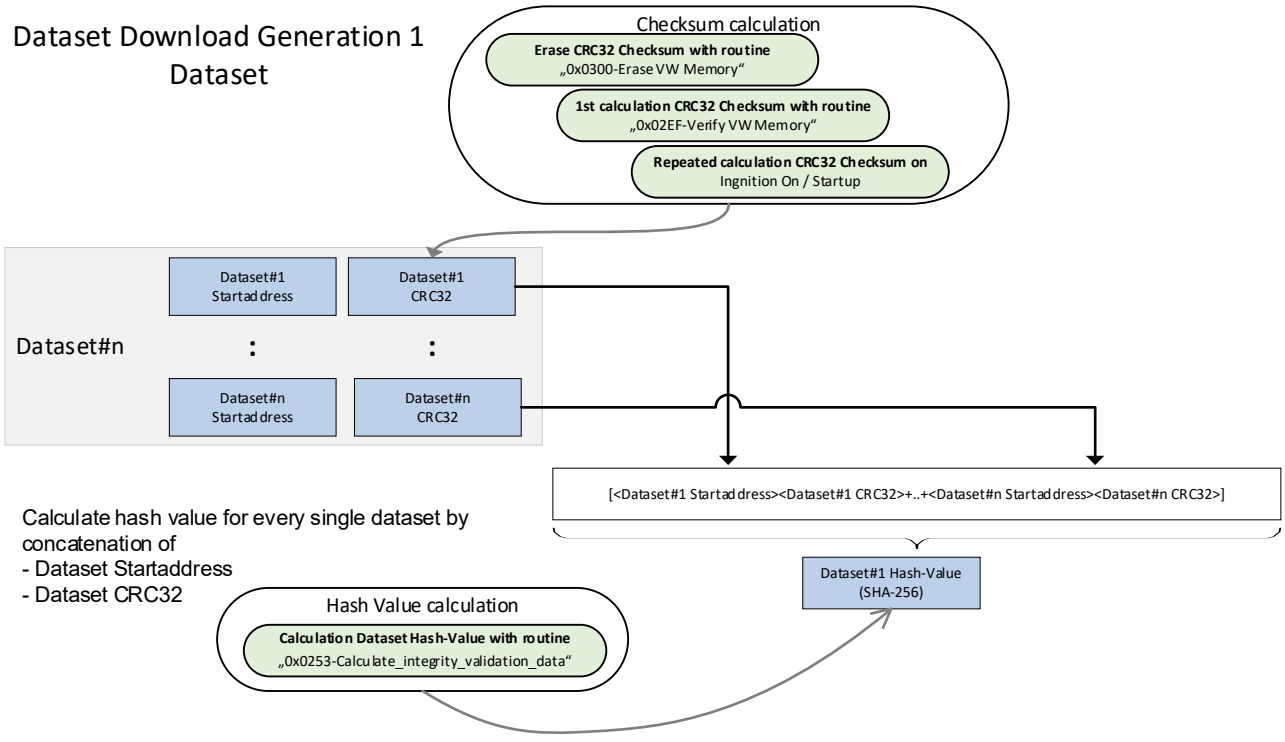
CRC32-Checksumme für einen Datensatz wird mit jedem Wechsel auf Kl.15-Ein oder Startup erneut berechnet und persistent gespeichert.

[allg. Anf.: F-LAH_RxSWIN-661]

Der Einzel-Hashwert für einen Datensatz wird mit dem Empfang des Requests für die Routine "0x0253-Calculate_integrity_validation_data" mit der ControlOption "Type_of_calculation" = 0x00 berechnet.

[I: F-LAH_RxSWIN-583]

Abbildung 4-6 CRC32 Checksummenberechnung über Datensätze des Datensatzdownload Generation 1



4.1.2.1.3 CRC32-Checksummenberechnung für Bootloader Datensätze der Datensatzdownload Generation 2 gemäß Dokument /7/

[allg. Anf.: F-LAH_RxSWIN-587]

Die CRC32-Checksumme eines Bootloader-Datensatzes wird mit dem Empfang des Requests für die Routine "0xFF00-Erase Memory", die den Bootloader-Datensatz adressiert, gelöscht.

[allg. Anf.: F-LAH_RxSWIN-714]

Die Routine "0xFF00-Erase Memory", die einen logischen Block mit Bootloader-Datensatz adressiert, darf den Programmierungs-Hashwert nicht beeinflussen.

[allg. Anf.: F-LAH_RxSWIN-588]

Die CRC32-Checksumme eines Bootloader-Datensatzes wird mit dem Empfang des Requests für die Routine "0x0202-Check Memory" erstmals berechnet.

[allg. Anf.: F-LAH_RxSWIN-589]

Die CRC32-Checksumme eines Bootloader-Datensatzes wird mit dem positiven Response für die Routine "0x0202-Check Memory" persistent gespeichert.

[allg. Anf.: F-LAH_RxSWIN-590]

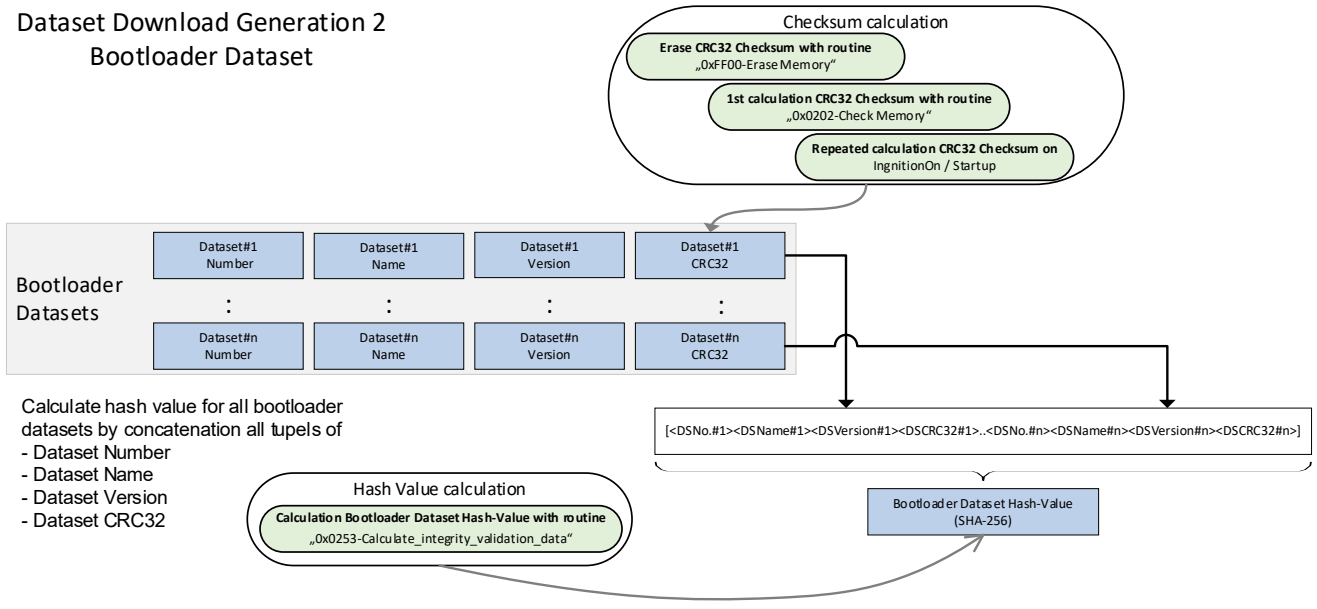
Die CRC32-Checksumme eines Bootloader-Datensatzes wird mit jedem Klemme 15 Wechsel oder Startup erneut berechnet und persistent gespeichert.

[allg. Anf.: F-LAH_RxSWIN-662]

Der Einzel-Hashwert für die Bootloader-Datensätze wird mit dem Empfang des Requests für die Routine "0x0253-Calculate_integrity_validation_data" mit der ControlOption "Type_of_calculation" = 0x00 berechnet.

[I: F-LAH_RxSWIN-591]

Abbildung 4-7 CRC32-Checksummenberechnung über Bootloader Datensätze der Generation 2



4.1.2.1.4 Gesamt-Hashwertberechnung der Konfigurationsdaten

[allg. Anf.: F-LAH_RxSWIN-347]

In DK2/DK2F/DK2FV/DK3/DK3V/DK4/DK4V/SWCL-Systemen wird anhand der Liste aus dem DataIdentifier "0x0250-Integrity_validation_data_configuration_list" der Hashwert der Konfigurationsdaten berechnet.

[allg. Anf.: F-LAH_RxSWIN-622]

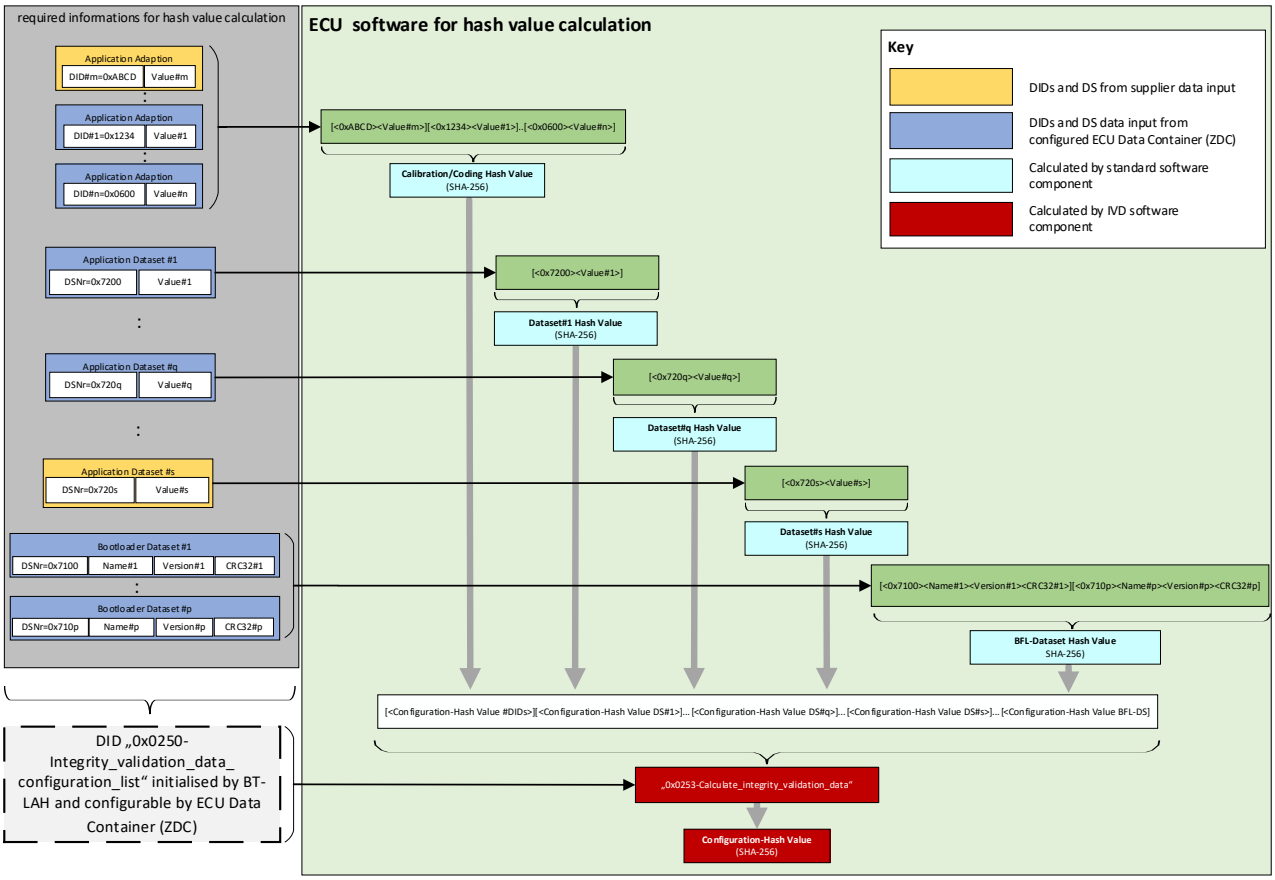
Die Routine "0x0253-Calculate_integrity_validation_data", mit der ControlOption "Type_of_calculation" = 0x00, stößt die Berechnung des Hashwertes der Konfigurationsdaten, über alle in der Liste aufgeführten DataIdentifier und Datensatznummern, in der durch die Liste vorgegeben Reihenfolge, an.

[allg. Anf.: F-LAH_RxSWIN-344]

Der auszugebende Gesamt-Hashwert aller Konfigurationsdaten ergibt sich als Hashwert über den Hashwert aller konkatenierten Einzelhashwerte.

[I: F-LAH_RxSWIN-236]

Abbildung 4-8 Berechnung des Gesamt-Hashwertes der Konfigurationsdaten (bei DSDL Gen. 2)

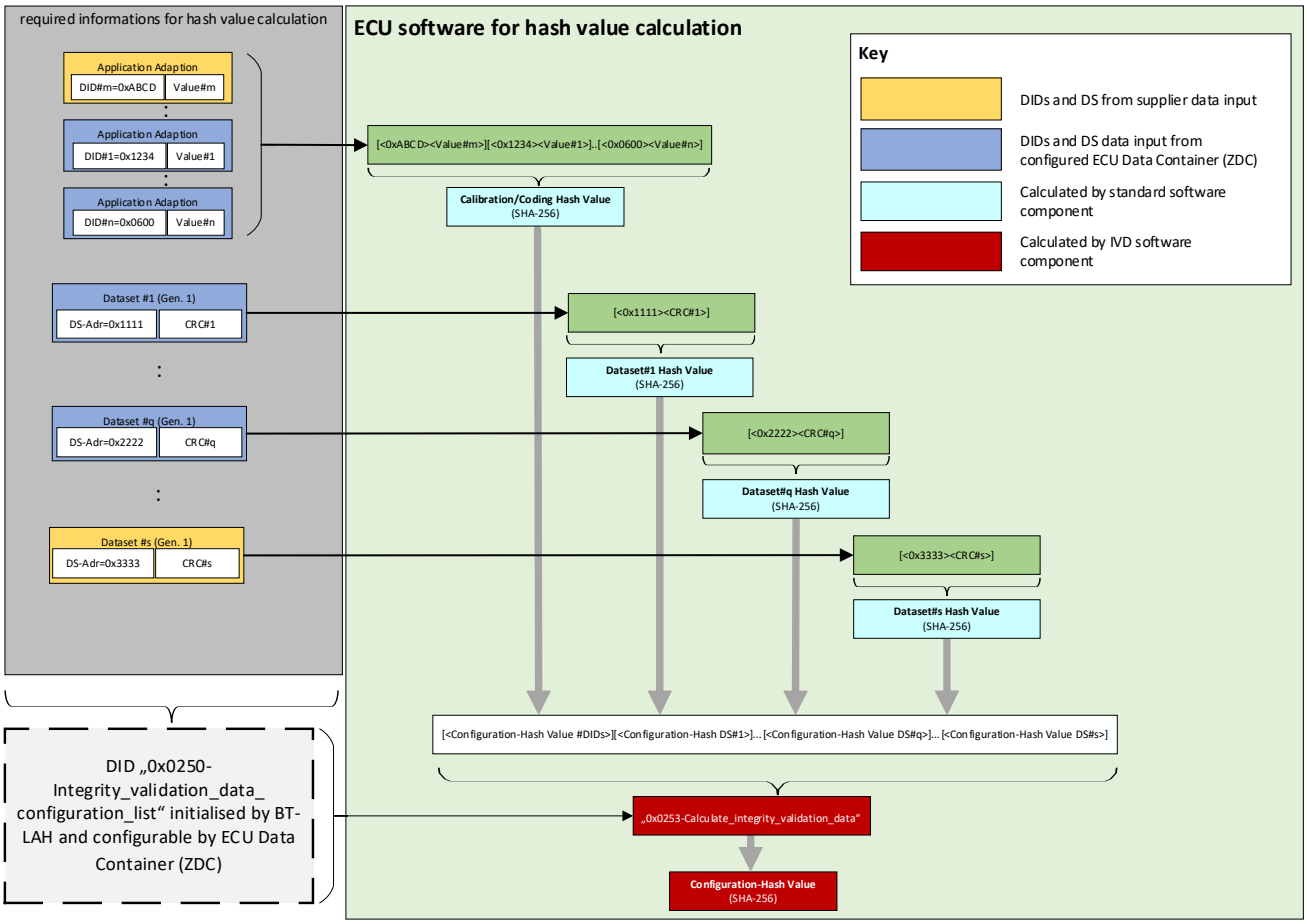


[I: F-LAH_RxSWIN-392]

DSNr = Datensatznummer (2 Byte Hex) gemäß Dokument /7/ 0x7100 - 0x71FF für Bootloader Datensätze und 0x7200 - 0x72FF für Applikations-Datensätze
DID = DataIdentifier gemäß Dokument /1/
Value = Dateninhalt der Anpassungen bzw. der Codierungen
Hash = Hashwert des Applikations-Datensatzes (SHA-256 mit 32 Byte Länge)
Version = Bootloader-Datensatz Version entsprechend Identifikation "0xF1B1-VW_Application_data_set_identification" mit der entsprechenden Datensatznummer gemäß Dokument /2/
Name = Bootloader-Datensatz Name entsprechend Identifikation "0xF1B3-VW_Data_set_name" mit der entsprechenden Datensatznummer gemäß Dokument /2/

[I: F-LAH_RxSWIN-666]

Abbildung 4-9 Berechnung des Gesamt-Hashwertes der Konfigurationsdaten (bei DSDL Gen. 1)



[I: F-LAH_RxSWIN-667]

DS-Adr = Datensatzadresse (2 Byte Hex) gemäß Dokument /11/
DID = DataIdentifier gemäß Dokument /1/
Value = Dateninhalt der Anpassungen bzw. der Codierungen
Hash = Hashwert des Applikations-Datensatzes (SHA-256 mit 32 Byte Länge)

4.1.2.2 Anforderungen an Prozesse

[Prozess-Anf.: F-LAH_RxSWIN-663]

Alle zur Konfigurations-Hashwert Berechnung relevanten Konfigurationsdaten müssen in den Diagnosedaten-Tabellen des BT-LAH gekennzeichnet werden und im ZDC enthalten sein.

[Prozess-Anf.: F-LAH_RxSWIN-538]

Die Initialbedatung des DataIdentifier "0x0250-Integrity_validation_data_configuration_list" erfolgt über die Diagnosedaten-Tabellen des BT-LAH.

[Prozess-Anf.: F-LAH_RxSWIN-560]

Die Initialbedatung des DataIdentifier "0x0250-Integrity_validation_data_configuration_list" enthält alle als ZDC-relevant gekennzeichneten DataIdentifier für Anpassungen und Codierung sowie die Datensatznummern der Applikations- und Bootloader-Datensätze.

[Prozess-Anf.: F-LAH_RxSWIN-375]

Der Inhalt des DataIdentifier "0x0250-Integrity_validation_data_configuration_list" kann bei Bedarf geändert werden, wenn die Initialbedatung von der Bedatung abweicht, die durch einen, mittels Bauauftrag konfigurierten, ZDC ermittelt wird.

[Prozess-Anf.: F-LAH_RxSWIN-771]

Die Berechnung des Soll-Konfigurations-Hashwert erfolgt im IT-System analog der Berechnung im Diagnose-Server.

[Prozess-Anf.: F-LAH_RxSWIN-479]

Die Reihenfolge der Werte der DataIdentifier und Datensatznummern im DataIdentifier "0x0250-Integrity_validation_data_configuration_list" ergibt sich aus dem, durch den Bauauftrag konfigurierten, ZDC.

4.2 Anforderungen an DK4/DK4V-Systeme auf Basis Q-LAH 80127 ab Version 5.1

[allg. Anf.: F-LAH_RxSWIN-200]

Für jedes DK2/DK2F/DK2FV-System muss das DK4-low/DK4V-low-System abhängig von der Trigger-Bedingung für Sammel-DIDs entsprechend Dokument /3/ die Identifikationsdaten erheben.

[allg. Anf.: F-LAH_RxSWIN-693]

Neben den Trigger-Bedingungen entsprechend Dokument /3/ ist zusätzlich folgende Trigger-Bedingung umzusetzen:

- Beenden des mittels SFD-E2E abgesicherten Schreibvorganges vom DK4-low-System zum DK2F-System.

4.2.1 Sammel-DataIdentifier für Integrity Validation Data der Programmierung von DK2F/DK2FV-Systemen

[I: F-LAH_RxSWIN-187]

Der DataIdentifier "0x0247-Slave_list_programming_hash" (Hashwerte der Programmierung von SubbusSystemen) beinhaltet die Integrity Validation Data über Instruction-Code aller unterlagerten DK2F/DK2FV-Systeme.

[allg. Anf.: F-LAH_RxSWIN-196]

Integrity Validation Data für Programmierung der DK2F/DK2FV-Systeme müssen vom DK4-low/DK4V-low-System mittels Service ReadDataByIdentifier (22hex) über den Sammel-DataIdentifier "0x0247-Slave_list_programming_hash" ausgegeben werden.

[allg. Anf.: F-LAH_RxSWIN-195]

Ein übergeordnetes DK4-low/DK4V-low-System sammelt für alle ihm untergeordneten DK2F/DK2FV-Systeme die Hashwerte über den Service ReadDataByIdentifier (22hex) mit dem DataIdentifier "0x0249-Programming_hash" ein und bildet über die gesammelten Hashwerte der unterlagerten Systeme den Gesamt-Hashwert.

[allg. Anf.: F-LAH_RxSWIN-197]

Der Sammel-DataIdentifier "0x0247-Slave_list_programming_hash" darf mittels Service WriteDataByIdentifier (2Ehex) nicht schreibbar sein.

[allg. Anf.: F-LAH_RxSWIN-199]

Die Implementierung des DataIdentifier "0x0247-Slave_list_programming_hash" ist verbindlich für alle DK4-low/DK4V-low-Systeme mit unterlagerten DK2F/DK2FV-Systemen.

4.2.2 Sammel-DataIdentifier für Integrity Validation Data der Konfiguration von DK2/DK2F/DK2FV-Systemen

[I: F-LAH_RxSWIN-480]

Der DataIdentifier "0x0248-Slave_list_configuration_hash" (Hashwerte der Konfiguration von SubbusSystemen) beinhaltet die Integrity Validation Data über die Konfigurationsdaten aller unterlagerten DK2/DK2F/DK2FV-Systeme.

[allg. Anf.: F-LAH_RxSWIN-127]

Integrity Validation Data für Konfigurationsdaten der DK2/DK2F/DK2FV-Systeme müssen vom DK4-low-System mittels Service ReadDataByIdentifier (22hex) über den Sammel-DataIdentifier "0x0248-Slave_list_configuration_hash" ausgegeben werden.

[allg. Anf.: F-LAH_RxSWIN-481]

Ein übergeordnetes DK4-low/DK4V-low-System sammelt für alle ihm unterlagerten DK2/DK2F/DK2FV-Systeme die Hashwerte, über den Service ReadDataByIdentifier (22hex) mit dem DataIdentifier "0x0245-Configuration_hash" ein und bildet über die gesammelten Hashwerte der unterlagerten Systeme den Gesamt-Hashwert.

[allg. Anf.: F-LAH_RxSWIN-129]

Der Sammel-DataIdentifier "0x0248-Slave_list_configuration_hash" darf mittels Service WriteDataByIdentifier (2Ehex) nicht schreibbar sein.

[allg. Anf.: F-LAH_RxSWIN-482]

Die Implementierung des Sammel-DataIdentifier "0x0248-Slave_list_configuration_hash" ist verbindlich für alle DK4-low/DK4V-low-Systeme mit unterlagerten DK2/DK2F/DK2FV-Systemen.

4.3 Anforderungen an DK4-Systeme auf Basis Q-LAH 80127 bis Version 4.0

[I: F-LAH_RxSWIN-425]

Für DK4-Systeme bis 80127 v4.0 (ohne Sammel-DataIdentifier) sind für Integration Validation Data von DK2-Systemen systemspezifische DataIdentifier in einem separaten DataIdentifier-Bereich reserviert.

[I: F-LAH_RxSWIN-735]

Beispiel zur systemspezifischen Bestimmung des Identifikationsservice (DataIdentifier und ODX-Bezeichnung):

DID-Wertebereich für Programmierungs-Hashwert: 0xA800 - 0xA9FF
SubSystemNode (SSN): 0x01
Longname Teil 1: Control_unit_for_wiper_motor
Trennzeichen: "_"
Longname Teil 2: Programming_hash

ODX-Bildungsregel:
Longnamekombination Teil 1 + Trennzeichen + Longname Teil 2 =
"Control_unit_for_wiper_motor_Programming_hash"

DID-Bildungsregel:
Offset 0xA800 + SSN =
"0xA801"

Hinweis: Diese Bildungsregeln gelten auch für den Konfigurations-Hashwert.

4.3.1 Programmierungs-Hashwert

[allg. Anf.: F-LAH_RxSWIN-399]

DK4-low-Systeme mit unterlagerten DK2F-Systemen müssen den Identifikationsservice "Slave_x_programming_hash" unterstützen.

[allg. Anf.: F-LAH_RxSWIN-484]

Der Wert des konkreten Identifikationsservice "Slave_x_programming_hash" ergibt sich für jedes DK2F-System mit einer SSN kleiner/gleich 0x1FF wie folgt: DataIdentifier 0xA800 + SubSystemNodeAddress (SSN).

[allg. Anf.: F-LAH_RxSWIN-400]

DK4-Systeme müssen die Daten der "Slave_x_programming_hash"-DIDs bei den unterlagerten DK2F-Systemen über den standardisierten DataIdentifier "0x0249-Programming_hash" abfragen.

[allg. Anf.: F-LAH_RxSWIN-485]

Bei DK2F-Systemen mit einer SSN kleiner/gleich 0x1FF kann die Abfrage auch über die individuellen DataIdentifier erfolgen.

[I: F-LAH_RxSWIN-488]

Hinweis: Bei DK2F-Systemen mit einer SSN größer 0x1FF müssen zwingend Sammel-DataIdentifier im DK4-Low-System verwendet werden.

[allg. Anf.: F-LAH_RxSWIN-529]

Der Identifikationsservice "Slave_x_programming_hash" ist nur lesbar und darf mittels Service WriteDataByIdentifier (2Ehex) nicht schreibbar sein.

4.3.2 Konfigurations-Hashwert

[allg. Anf.: F-LAH_RxSWIN-396]

DK4-low-Systeme mit unterlagerten DK2/DK2F-Systemen müssen den Identifikationsservice "Slave_x_configuration_hash" unterstützen.

[allg. Anf.: F-LAH_RxSWIN-486]

Der Wert des konkreten Identifikationsservice "Slave_x_configuration_hash" ergibt sich für jedes DK2/DK2F-System mit einer SSN kleiner/gleich 0x1FF wie folgt: DataIdentifier 0xAA00 + SubSystemNodeAddress (SSN).

[allg. Anf.: F-LAH_RxSWIN-397]

DK4-low-Systeme müssen die Daten der "Slave_x_configuration_hash"-DIDs bei den unterlagerten DK2/DK2F-Systemen über den standardisierten DataIdentifier "0x0245-Configuration_hash" abfragen.

[allg. Anf.: F-LAH_RxSWIN-487]

Bei DK2/DK2F-Systemen mit einer SSN kleiner/gleich 0x1FF kann die Abfrage auch über die individuellen DataIdentifier erfolgen.

[I: F-LAH_RxSWIN-489]

Hinweis: Bei DK2/DK2F-Systemen mit einer SSN größer 0x1FF müssen zwingend Sammel-DataIdentifier im DK4-Low-System verwendet werden.

[allg. Anf.: F-LAH_RxSWIN-528]

Der Identifikationsservice "Slave_x_configuration_hash" ist nur lesbar und darf mittels Service WriteDataByIdentifier (2Ehex) nicht schreibbar sein.

4.4 Anforderungen an DK2/DK2F/DK2FV-Systeme

4.4.1 Programmierungs-Hashwert

[I: F-LAH_RxSWIN-186]

Über den DataIdentifier "0x0249-Programming_hash" wird die Integrity Validation Data der Programmierung eines DK2F/DK2FV-Servers ausgegeben.

[allg. Anf.: F-LAH_RxSWIN-190]

Der DataIdentifier "0x0249-Programming_hash" ist nur lesbar und darf mittels Service WriteDataByIdentifier (2Ehex) nicht schreibbar sein.

[allg. Anf.: F-LAH_RxSWIN-191]

Das Starten der Routine "0xFF00-Erase Memory", die einen logischen Block mit Instruction-Code adressiert, entsprechend Dokument /6/ erfordert das Löschen der bestehenden Integrity Validation Data im DK2F/DK2FV-System für den DataIdentifier "0x0249-Programming_hash".

[allg. Anf.: F-LAH_RxSWIN-744]

Der Empfang des Requests mit dem Service ReadDataByIdentifier (22hex) zum Lesen des DataIdentifier "0x0249-Programming_hash" des Hashwertes eines DK2F/DK2FV Diagnose-Servers bewirkt eine Neuberechnung des Hashwertes in dem DK2F/DK2FV-System. Solange die Neuberechnung nicht abgeschlossen ist, muss der Request mit einem Response-Pending (NRC 0x78) negativ beantwortet werden. Erst nach dem Abschluss der Neuberechnung wird der Request positiv beantwortet.

[allg. Anf.: F-LAH_RxSWIN-738]

In DK2F/DK2FV-System mit einer SSN kleiner oder gleich 0x1FF, erfordert das Starten der Routine "0xFF00-Erase Memory", die einen logischen Block mit Instruction-Code adressiert, entsprechend Dokument /6/ das Löschen der bestehenden Integrity Validation Data in dem individuelle DataIdentifier 0xA800 + SSN "VW_slave_programming_hash".

[allg. Anf.: F-LAH_RxSWIN-626]

Mit dem Starten der Routine "0xFF00-Erase Memory" für einen logischen Block mit Instruction-Code, entsprechend Dokument /6/, muss die CRC32-Checksumme des adressierten Blocks gelöscht werden.

[allg. Anf.: F-LAH_RxSWIN-627]

Nach der positiven Prüfung, dass der logische Block mit der Routine "0x0202-Check Memory" entsprechend Dokument /6/ fehlerfrei übertragen wurde, muss die CRC32-Checksumme des logischen Blocks neu berechnet werden.

[allg. Anf.: F-LAH_RxSWIN-719]

Mit dem positiven Response für die Routine "0x0202-Check Memory", die einen logischen Block mit Instruction-Code adressiert, wird die berechnete CRC32-Checksumme des logischen Blocks persistent gespeichert.

[allg. Anf.: F-LAH_RxSWIN-490]

Zur Gewährleistung der Abwärtskompatibilität der DK2F/DK2FV-System mit einer SSN kleiner/gleich 0x1FF zu DK4-Systemen auf Basis VW80127 V4.0 muss zusätzlich zu dem DataIdentifier "0x0249-Programming_hash" auch der individuelle DataIdentifier aus dem Bereich 0xA800 bis 0xA9FF mit gleichem Inhalt/Funktionalität unterstützt werden.

[I: F-LAH_RxSWIN-736]

Hinweis: Der generische DataIdentifier "0x0249-Programming_hash" im DK2F/DK2FV-System erfordert den Sammel-DataIdentifier "0x0247-Slave_list_programming_hash" im übergeordneten DK4-low-System.

4.4.2 Konfigurations-Hashwert

[I: F-LAH_RxSWIN-491]

Über den DataIdentifier "0x0245-Configuration_hash" wird die Integrity Validation Data der Konfigurationsdaten eines DK2/DK2F/DK2FV-Servers ausgegeben.

[allg. Anf.: F-LAH_RxSWIN-130]

Der DataIdentifier "0x0245-Configuration_hash" ist nur lesbar und darf mittels Service WriteDataByIdentifier (2Ehex) nicht schreibbar sein.

[allg. Anf.: F-LAH_RxSWIN-138]

Der Empfang des Requests mit dem Service ReadDataByIdentifier (22hex) zum Lesen des DataIdentifier "0x0245-Configuration_hash" des Hashwertes eines DK2/DK2F/DK2FV Diagnose-Servers bewirkt eine Neuberechnung des Hashwertes in dem DK2/DK2F/DK2FV-System. Solange die Neuberechnung nicht abgeschlossen ist, muss der Request mit einem Response-Pending (NRC 0x78) negativ beantwortet werden. Erst nach dem Abschluss der Neuberechnung wird der Request positiv beantwortet.

[allg. Anf.: F-LAH_RxSWIN-739]

In DK2F/DK2FV-System mit einer SSN kleiner oder gleich 0x1FF, erfordert der empfangene Request mit dem Service ReadDataByIdentifier (22hex) zum Lesen des individuellen DataIdentifier 0xAA00 + SSN "VW_slave_configuration_hash" die Neuberechnung des Integrity Validation Data.

[I: F-LAH_RxSWIN-140]

Der Hashwert wird über alle in der Liste des DataIdentifier "0x0250-Integrity_validation_data_configuration_list" enthaltenen schreibbaren DataIdentifier berechnet. Diese Berechnung ist detailliert im Abschnitt "Integrity Validation Data / Allgemeine Anforderungen / Konfigurations-Hashwert" beschrieben.

[allg. Anf.: F-LAH_RxSWIN-492]

Zur Gewährleistung der Abwärtskompatibilität der DK2F/DK2FV-System mit einer SSN kleiner/gleich 0x1FF zu DK4-Systemen auf Basis VW80127 V4.0 muss zusätzlich zu dem DataIdentifier"0x0245-Configuration_hash" auch der individuelle DataIdentifier aus dem Bereich 0xAA00 bis 0xABFF mit gleichem Inhalt/Funktionalität unterstützt werden.

[I: F-LAH_RxSWIN-737]

Hinweis: Der generische DataIdentifier "0x0245-Configuration_hash" im DK2F/DK2FV-System erfordert den Sammel-DataIdentifier "0x0248-Slave_list_configuration_hash" im übergeordneten DK4-low-System.

4.5 Standardsoftware-Modul für Integrity Validation Data

[I: F-LAH_RxSWIN-845]

Für die Implementierung von Integrity Validation Data werden von der VOLKSWAGEN AG verschiedene SSW-Module (AUTOSAR v4.3) und eine C-Referenzimplementierung zur Verfügung gestellt.

4.6 Ablauf

[I: F-LAH_RxSWIN-931]

Bei den folgenden Abbildungen handelt es sich um beispielhafte Teil-Abläufe, die nicht implementiert werden dürfen.

4.6.1 Beispielhaftes Auslesen aller relevanten Identifikationsdaten und Integrity Validation Data eines Diagnose-Servers

[I: F-LAH_RxSWIN-246]

Ein Diagnose-Client liest die Identifikationen eines DK4-Systems mit unterlagerten DK2/DK2F/DK2FV-Systemen aus.
Ausgelesen werden die:

[I: F-LAH_RxSWIN-836]

- Fahrzeug und Steuergeräte spezifischen Identifikationsdaten

[I: F-LAH_RxSWIN-835]

- Integrity Validation Data der Konfigurationsdaten der Anpassung/Codierung/Applikation- und Bootloader-Datensätze eines DK4-Systems

[I: F-LAH_RxSWIN-834]

- Integrity Validation Data der Programmierung (Instruction-Code) eines DK4-Systems

[I: F-LAH_RxSWIN-833]

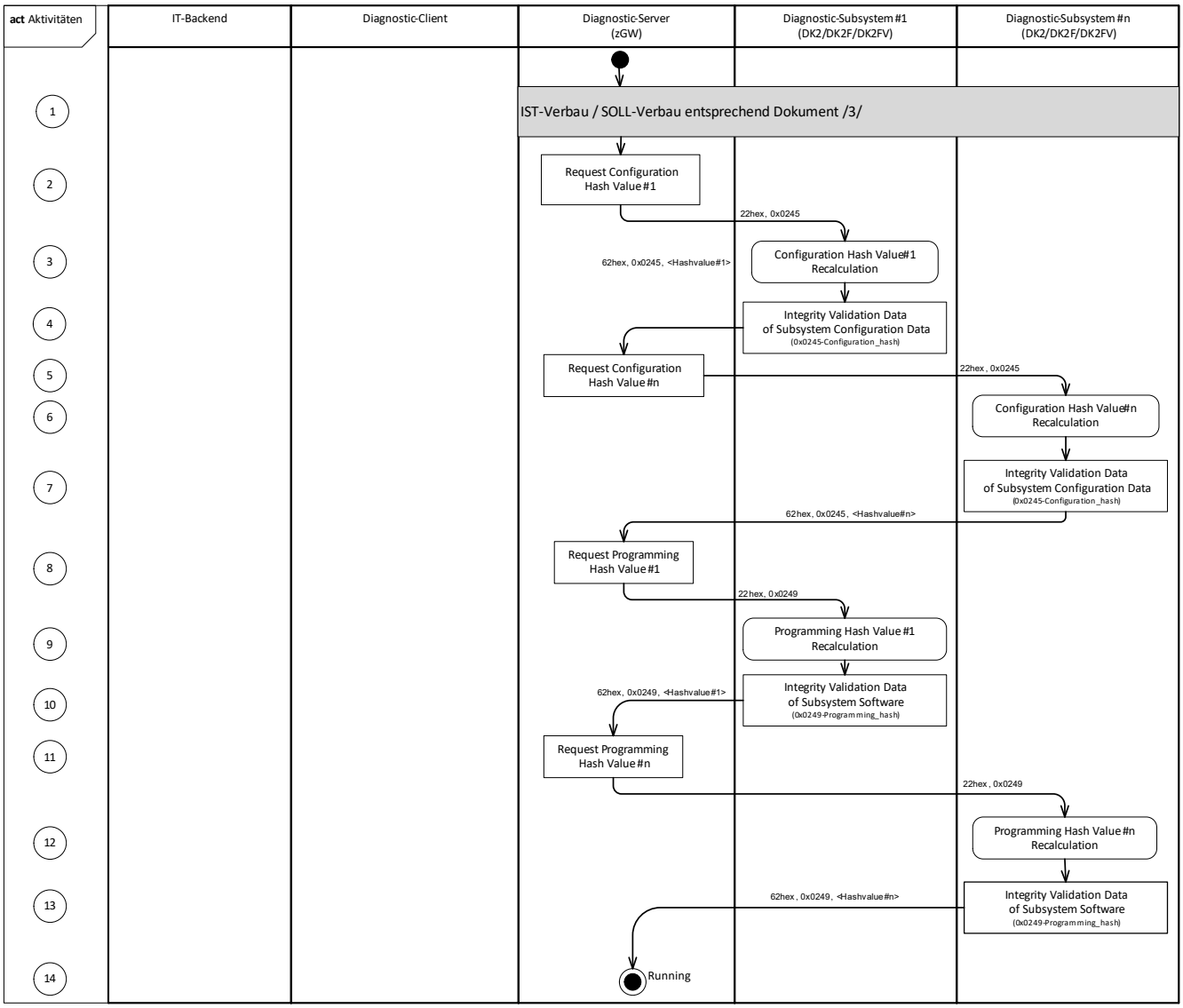
- Integrity Validation Data der Konfigurationsdaten der Anpassung (Data) eines unterlagerten DK2/DK2F/DK2FV-Systems

[I: F-LAH_RxSWIN-832]

- Integrity Validation Data der Programmierung (Instruction-Code) eines unterlagerten DK2F/DK2FV-Systems

[I: F-LAH_RxSWIN-670]

Abbildung 4-10 Initialisierung der Sammel-DataIdentifier für Integrity Validation Data am Beispiel Gateway - Teil 1/4



[I: F-LAH_RxSWIN-865]

1 – Nach Aufstarten des zGW aktualisiert das zGW die Ist-Verbauliste mit allen erkannten unterlagerten Diagnostic-Subsystemen nach Dokument /3/, Kapitel " Erhebung der Identifikationsdaten von DK1/2/2F-Systemen".

[I: F-LAH_RxSWIN-671]

2 – Nach der Aktualisierung der Verbauliste des zGWs fragt das zGW am unterlagerten Diagnostic-Subsystem#1 den DataIdentifier "0x0245-Configuration_hash“ für den Hashwert der Konfigurationsdaten ab.

[I: F-LAH_RxSWIN-672]

3 – In dem Diagnostic-Subsystem#1 startet der Empfang des Requests ReadDataByIdentifier mit dem DataIdentifier "0x0245-Configuration_hash" die Neuberechnung des Hashwertes der Konfigurationsdaten.

[I: F-LAH_RxSWIN-673]

4 – Das Diagnostic-Subsystem#1 sendet die Response mit dem neuberechneten Hashwert der Konfigurationsdaten. Das zGW aktualisiert den Sammel-DataIdentifier "0x0248-Slave_list_configuration_hash" anhand des empfangenen DataIdentifiers "0x0245-Configuration_hash".

[I: F-LAH_RxSWIN-674]

5 – Das zGW fragt am unterlagerten Diagnostic-Subsystem#n den DataIdentifier "0x0245-Configuration_hash“ für den Hashwert der Konfigurationsdaten des unterlagerten Diagnostic-Subsystems#n ab.
Hinweis: Das Abfragen der Diagnostic-Subsysteme #1 bis #n kann an den Bussen parallel erfolgen zur Zeitersparnis, hier wird nur die sequentielle Abfrage dargestellt.

[I: F-LAH_RxSWIN-675]

6 – In dem Diagnostic-Subsystem#n startet der Empfang des Requests ReadDataByIdentifier mit dem DataIdentifier "0x0245-Configuration_hash" die Neuberechnung des Hashwertes der Konfigurationsdaten.

[I: F-LAH_RxSWIN-676]

7 – Das Diagnostic-Subsystem#n sendet den neuberechneten Hashwert der Konfigurationsdaten. Das zGW aktualisiert den Sammel-DataIdentifier "0x0248-Slave_list_configuration_hash" anhand des empfangenen DataIdentifiers "0x0245-Configuration_hash".

[I: F-LAH_RxSWIN-677]

8 – Das zGW fragt am unterlagerten Diagnostic-Subsystem#1 den DataIdentifier "0x0249-Programming_hash" für den Hashwert der Programmierung ab.

[I: F-LAH_RxSWIN-710]

9 – In dem Diagnostic-Subsystem#1 startet der Empfang des Requests ReadDataByIdentifier mit dem DataIdentifier "0x0249-Programming_hash" die Neuberechnung des Hashwertes der Programmierungsdaten.

[I: F-LAH_RxSWIN-679]

10 – Das unterlagerten Diagnostic-Subsystem#1 sendet den gespeicherten Hashwert der Programmierung. Das zGW aktualisiert den Sammel-DataIdentifier "0x0247-Slave_list_configuration_hash" anhand des empfangenen DataIdentifiers "0x0249-Programming_hash".

[I: F-LAH_RxSWIN-680]

11 – Das zGW fragt am Diagnostic-Subsystem#n den DataIdentifier "0x0249-Programming_hash" für den Hashwert der Programmierung ab.

[I: F-LAH_RxSWIN-711]

12 – In dem Diagnostic-Subsystem#n startet der Empfang des Requests ReadDataByIdentifier mit dem DataIdentifier "0x0249-Programming_hash" die Neuberechnung des Hashwertes der Programmierungsdaten.

[I: F-LAH_RxSWIN-682]

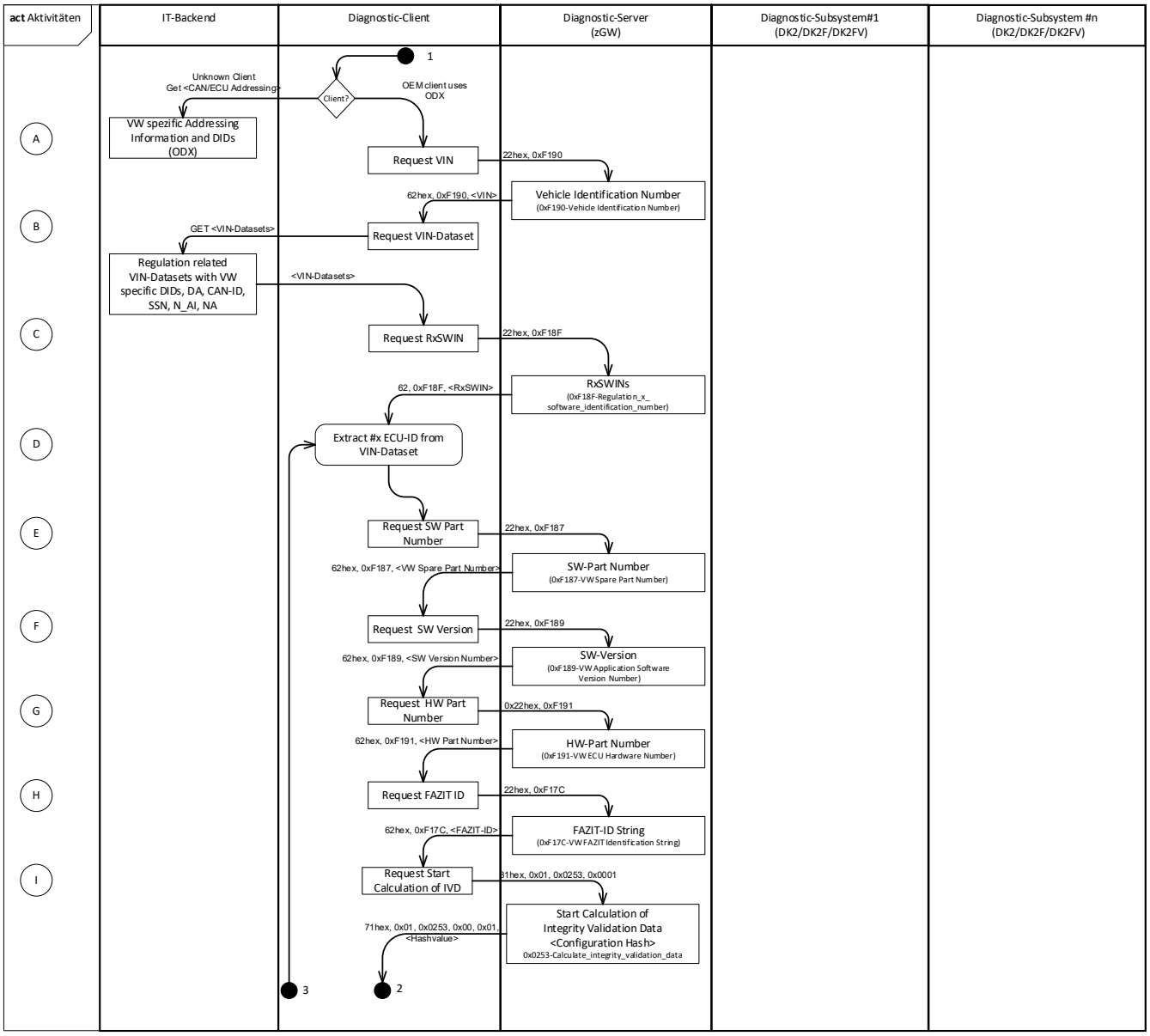
13 – Das Diagnostic-Subsystem#n sendet den gespeicherten Hashwert der Programmierung. Das zGW aktualisiert den Sammel-DataIdentifier "0x0247-Slave_list_programming_hash" anhand des empfangenen DataIdentifiers "0x0249-Programming_hash".

[I: F-LAH_RxSWIN-683]

14 – Das zGW aktualisiert die weiteren Sammel-DataIdentifier der Diagnostic-Subsysteme.

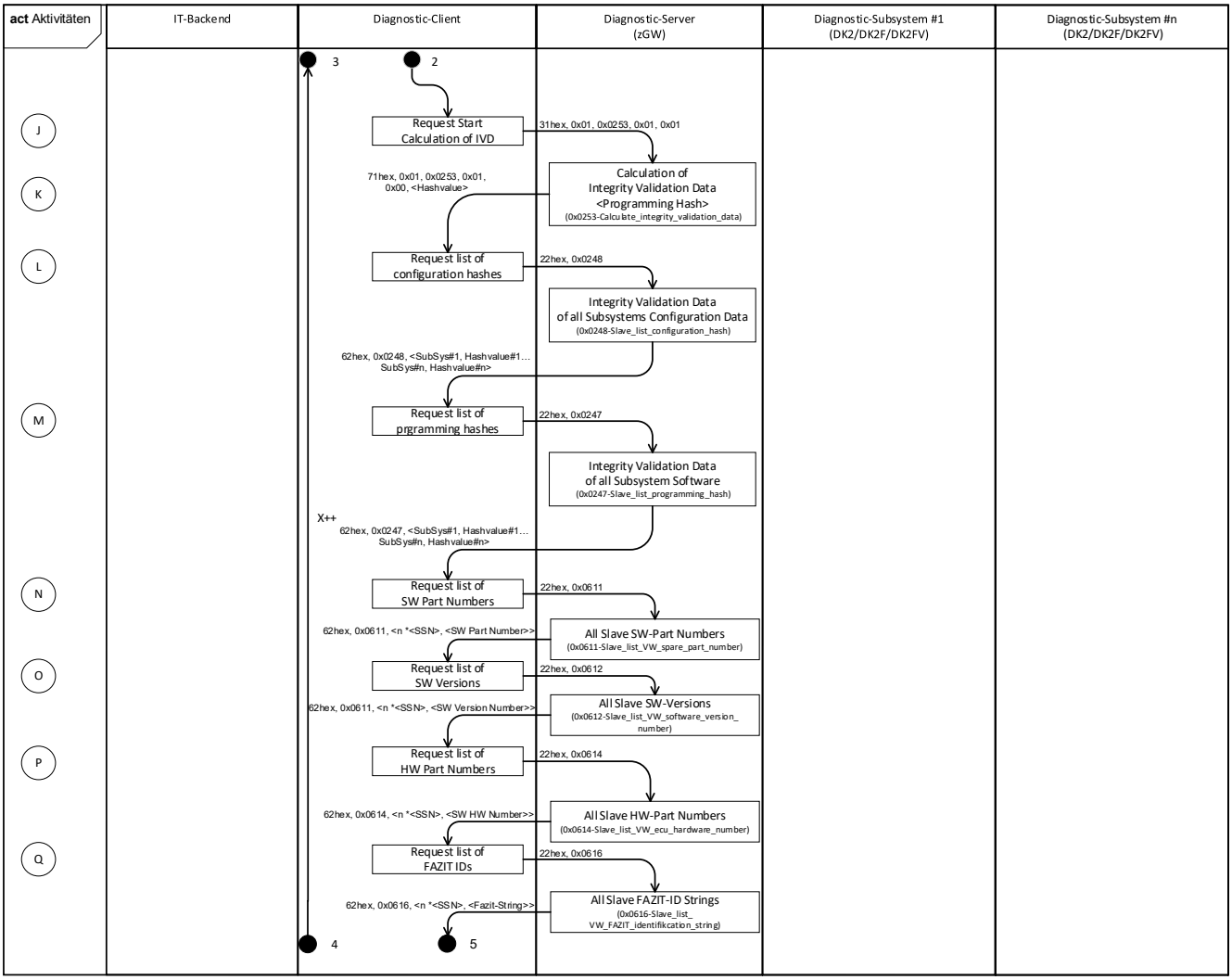
[I: F-LAH_RxSWIN-245]

Abbildung 4-11 Auslesen aller Identifikationsdaten und Integrity Validation Data - Teil 2/4



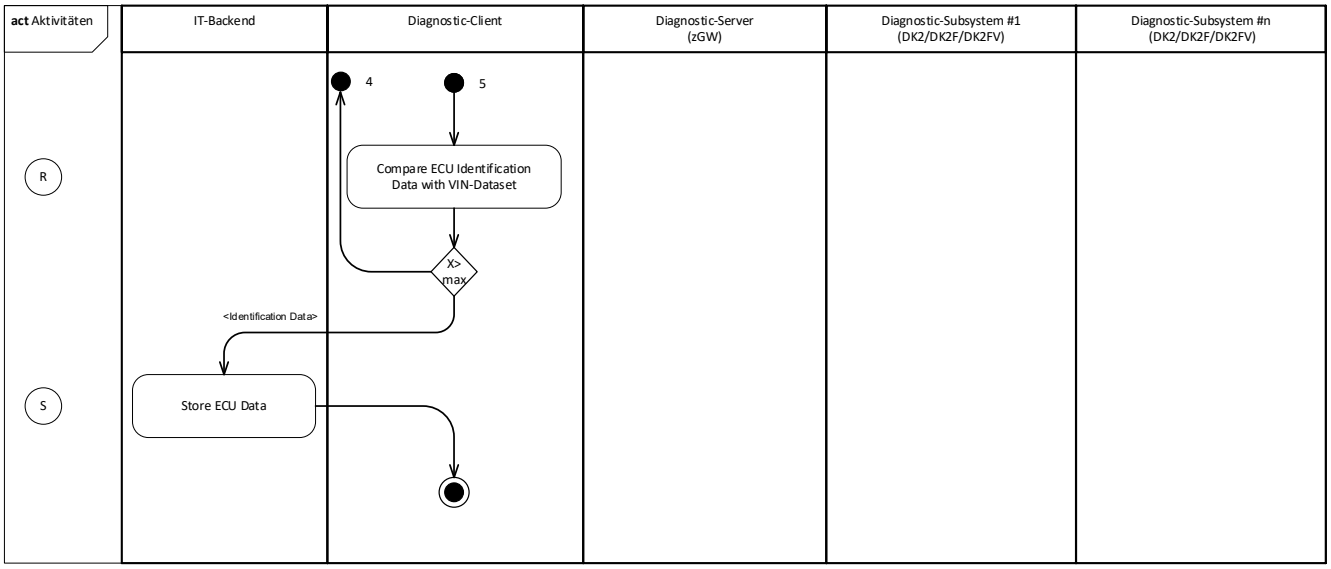
[I: F-LAH_RxSWIN-248]

Abbildung 4-12 Auslesen aller Identifikationsdaten und Integrity Validation Data - Teil 3/4



[I: F-LAH_RxSWIN-250]

Abbildung 4-13 Auslesen aller Identifikationsdaten und Integrity Validation Data Teil 4/4



[I: F-LAH_RxSWIN-249]

A – Der Diagnostic-Client fragt die notwendigen Adressierungsinformationen zur Kommunikation mit dem Fahrzeug und die notwendigen DataIdentifier (DID) für die Identifikationsdaten des Fahrzeugs vom IT-Backend ab. Adressierungsinformationen sind die Diagnose-Adressen (DA), Knoten-Adressen (NA), Subsystem-Knotenadresse (SSN), Netzwerkadress-Information (N_AI), CAN-Identifier für Request und Response. Der Diagnostic-Client fragt die Fahrgestellnummer des aktuellen Fahrzeugs über den DataIdentifier "0xF190-Vehicle Identification Number" ab. Bei Verwendung von O-DIS stehen alle notwendigen Informationen in den ODX-Daten zur Verfügung.

[I: F-LAH_RxSWIN-251]

B – Der Diagnostic-Client fragt den fahrzeugspezifischen und regulierungsbezogenen Datensatz für die Fahrgestellnummer des Fahrzeugs am IT-Backend ab, um weiter Adressierungsinformationen, wie z. B. Diagnose-Adressen, Knoten-Adressen und DataIdentifier zu erhalten.

[I: F-LAH_RxSWIN-252]

C – Der Diagnostic-Client fragt den DataIdentifier "0xF18F-Regulation_x_software_identification_numbers" mit der Liste aller Regulierungsnummern und SW-Identifikationsnummer am zentralen Gateway ab.

[I: F-LAH_RxSWIN-253]

D – Der Diagnostic-Client adressiert anhand der Daten des regulierungsbezogenen Datensatz nacheinander die einzelnen im Fahrzeug verbauten Diagnostic-Server.

[I: F-LAH_RxSWIN-254]

E – Der Diagnostic-Client liest die SW-Teilnummer über den DataIdentifier "0xF187-VW Spare Part Number" des adressierten Diagnostic-Server aus.

[I: F-LAH_RxSWIN-255]

F – Der Diagnostic-Client liest die SW-Version über den DataIdentifier "0xF189-VW Software Version Number" des adressierten Diagnostic-Server aus.

[I: F-LAH_RxSWIN-256]

G – Der Diagnostic-Client liest die HW-Teilnummer über den DataIdentifier "0xF191-VW ECU Hardware Number" des adressierten Diagnostic-Server aus.

[I: F-LAH_RxSWIN-257]

H – Der Diagnostic-Client liest die FAZIT-ID über den DataIdentifier "0xF17C-VW FAZIT Identification String" des adressierten Diagnostic-Server aus.

[I: F-LAH_RxSWIN-259]

I – Der Diagnostic-Client fragt den Hashwert der Konfigurationsdaten des adressierten Diagnostic-Server ab. Der DataIdentifier "0x0250-Integrity_validation_data_configuration_list" enthält dafür die Liste aller DataIdentifier und Datensatznummern über die der Hashwert der Konfigurationsdaten berechnet werden muss. Der RoutineIdentifier "0x0253-Calculate_integrity_validation_data" liefert in der positiven Response den Hashwert der Konfigurationsdaten des Diagnostic-Server.

[I: F-LAH_RxSWIN-260]

J – Der Diagnostic-Client startet für den adressierten Diagnostic-Server die Berechnung des Hashwertes über die gesamte Programmierung.

[I: F-LAH_RxSWIN-262]

K – Der Diagnostic-Client erhält für den Diagnostic-Server den neu berechneten Hashwert der Programmierung.

[I: F-LAH_RxSWIN-263]

L – Der Diagnostic-Client fragt für den Diagnostic-Server den Sammel-DataIdentifier "0x0248-Slave_list_configuration_hash", für die Hashwerte der Konfigurationsdaten aller Diagnostic-Subsysteme ab.

[I: F-LAH_RxSWIN-264]

M – Der Diagnostic-Client fragt für den Diagnostic-Server den Sammel-DataIdentifier "0x0247-Slave_list_programming_hash", für die Hashwerte der Programmierung aller Diagnostic-Subsysteme ab.

[I: F-LAH_RxSWIN-266]

N – Der Diagnostic-Client liest die Liste aller SW-Teilnummer der Diagnostic-Subsysteme über den Sammel-DataIdentifier "0x0611-Slave_list_VW_spare_part_number" über den adressierten Diagnostic-Server aus.

[I: F-LAH_RxSWIN-545]

O – Der Diagnostic-Client liest die Liste aller SW-Version der Diagnostic-Subsysteme über den Sammel-DataIdentifier "0x0612-Slave_list_VW_software_version_number" über den adressierten Diagnostic-Server aus.

[I: F-LAH_RxSWIN-546]

P – Der Diagnostic-Client liest die Liste aller HW-Teilnummer der Diagnostic-Subsysteme über den Sammel-DataIdentifier "0x0614-Slave_list_VW_ecu_hardware_number" über den adressierten Diagnostic-Server aus.

[I: F-LAH_RxSWIN-267]

Q – Der Diagnostic-Client liest die Liste aller FAZIT-Identifikationen der Diagnostic-Subsysteme über den Sammel-DataIdentifier "0x0616-Slave_list_VW_FAZIT_identification_string" über den adressierten Diagnostic-Server aus.

[I: F-LAH_RxSWIN-270]

R – Diagnostic-Client vergleicht die Identifikationsdaten für die RxSWIN mit dem Datensatz aus dem IT-Backend und liest danach die Identifikationsdaten für das nächste Diagnostic-Server aus.

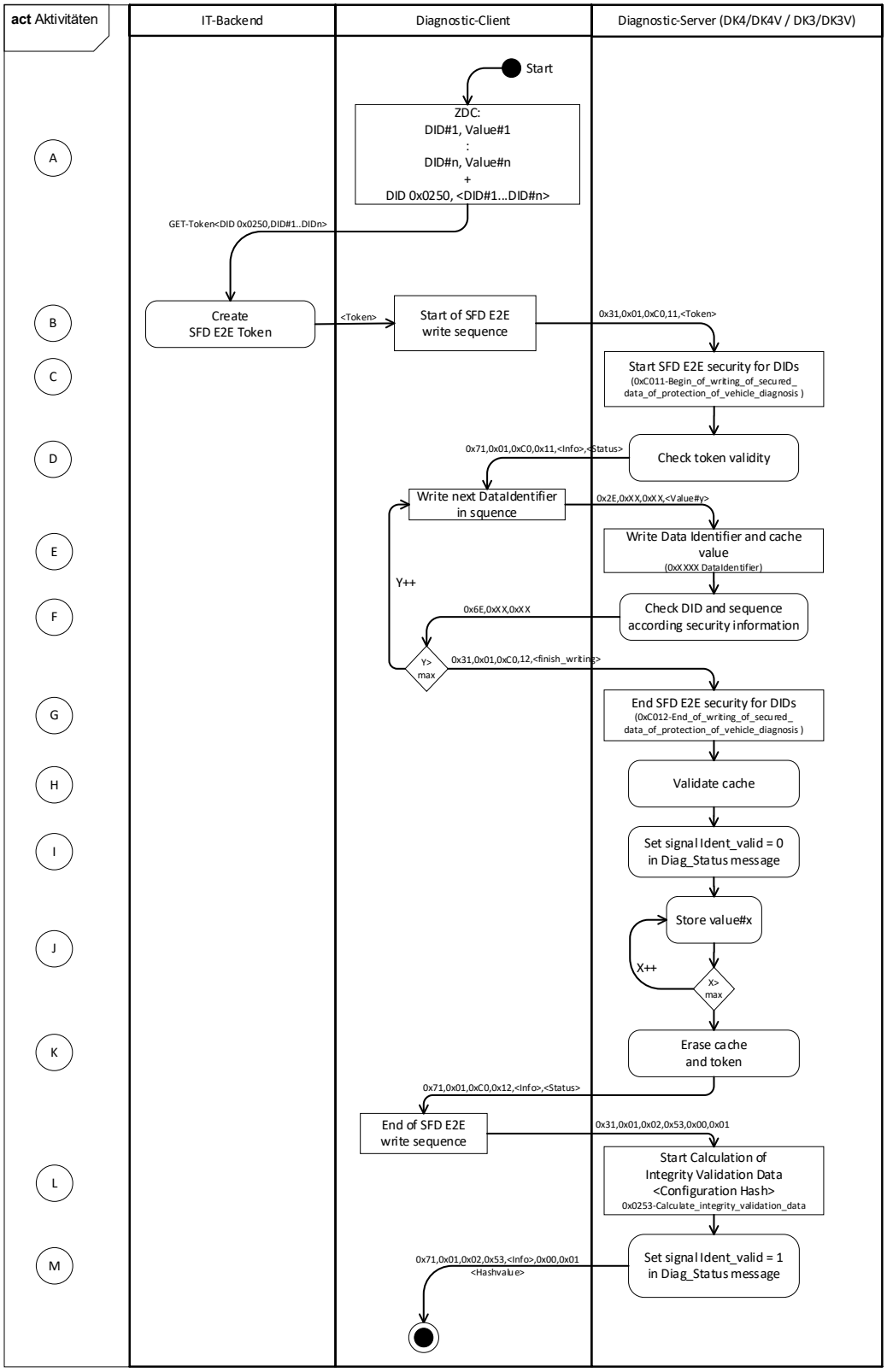
[I: F-LAH_RxSWIN-547]

S – Der Diagnostic-Client speichert die ausgelesenen Identifikationsdaten im IT-Backend.

4.6.2 Beispielhaftes Schreiben von Daten in ein SFD-E2E abgesichertes DK4/DK4V bzw. DK3/DK3V System

[I: F-LAH_RxSWIN-356]

Abbildung 4-14 Schreiben von Daten eines DK3/DK3V bzw. DK4/DK4V-Systems, Teil 1/1



[I: F-LAH_RxSWIN-537]

A - Die für die Steuergeräte-Konfiguration notwendigen DataIdentifier werden dem Diagnostic-Client zur Verfügung gestellt. Die DataIdentifier ergeben sich durch den Bauauftrag, der den ZDC konfiguriert, hieraus resultiert auch der Inhalt des DataIdentifier "0x0250-Integrity_validation_data_configuration_list". Der Diagnostic-Client fragt ein Absicherungs-Token am SFD-IT-Backend für die im konfigurierten ZDC enthaltene DataIdentifier DID#1 ... DID#n und dem berechneten DataIdentifier "0x0250-Integrity_validation_data_configuration_list" an.

[I: F-LAH_RxSWIN-359]

B* - Das SFD-IT-Backend erstellt ein Absicherungs-Token über die DataIdentifiers DID#1 ... DID#n und DID 0x0250.

[I: F-LAH_RxSWIN-360]

C* - Der Absicherungs-Token wird mit dem Start der SFD-Routine "0xC011-Begin_of_writing_of_secured_data_of_protection_of_vehicle_diagnosis" an den Diagnostic-Server übertragen und leitet die Schreib-Sequenz für SFD-Ende-zu-Ende abgesicherte Daten ein.

[I: F-LAH_RxSWIN-361]

D* - Der Diagnostic-Server prüft die Validität des empfangenen SFD-Absicherungs-Tokens. Entsprechend der SFD-E2E Absicherung gemäß Dokument /4/ wird bei einem Empfang eines ungültigen Tokens die Absicherungsinformationen gelöscht.

[I: F-LAH_RxSWIN-362]

E - Die DataIdentifier aus dem konfigurierten ZDC und der DataIdentifier "0x0250-Integrity_validation_data_configuration_list**"** werden mittels Service WriteDataByIdentifier (2Ehex) an den Diagnostic-Server übertragen und zwischengespeichert. Ohne SFD werden die Daten direkt in den Speicher geschrieben.

[I: F-LAH_RxSWIN-363]

F* - Entsprechend der SFD-E2E Absicherung aus Dokument /4/ wird die Empfangsreihenfolge der einzelnen DataIdentifier entsprechend der Absicherungsinformation überprüft. Bei positiver Prüfung antwortet der Diagnostic-Server mit einem positiven Response. Bei einem negativen Prüfergebnis werden die empfangen DataIdentifier und die zugehörigen DataRecords sowie die Absicherungsinformationen gelöscht. Bei einem negativen Prüfergebnis antwortet der Diagnostic-Server mit einem NRC 0x22 (ConditionsNotCorrect).

[I: F-LAH_RxSWIN-364]

G* - Die SFD-Schreibsequenz wird mit dem Start der SFD-Routine "0xC012-End_of_writing_of_secured_data_of_protection_of_vehicle_diagnosis" mit RoutineControlOption [0x01-Finish_writing_of_secured_data] abgeschlossen.

[I: F-LAH_RxSWIN-365]

H* - Die Authentizität der übertragenen DataIdentifier wird anhand der Absicherungsinformation geprüft.

[I: F-LAH_RxSWIN-368]

I - In der Diag_Status Botschaft des Diagnostic-Server wird im Signal [Ident_valid] der Wert '0' angezeigt, dass die Identifikationsdaten aktuell nicht vollständig gültig sind, d. h. die Hashwerte der Konfigurationsdaten sind gelöscht und sind noch nicht erneut berechnet worden.

[I: F-LAH_RxSWIN-366]

J* - Bei positiver Prüfung werden alle zwischengespeicherten DataIdentifier aus dem Zwischenspeicher sequentiell in den Zielspeicher des Diagnostic-Server geschrieben. Bei einem negativen Prüfergebnis werden alle empfangen DataIdentifier und die zugehörigen DataRecords sowie die Absicherungsinformationen gelöscht.

[I: F-LAH_RxSWIN-369]

K* - Der Zwischenspeicher und die Absicherungsinformationen im dem Diagnostic-Server werden gelöscht, nachdem alle DataIdentifier in den Zielspeicher übertragen wurden.

[I: F-LAH_RxSWIN-370]

L - Der Diagnostic-Client fragt den neuen Hashwert der Konfigurationsdaten ab. Der Aufruf der Routine "0x0253-Calculate_integrity_validation_data" startet die Neuberechnung des Hashwertes der Konfigurationsdaten auf dem Diagnostic-Server. Die Hashwertberechnung erfolgt entsprechend der im DataIdentifier "0x0250-Integrity_validation_data_configuration_list**"** (siehe Kapitel "0x0250-Integrity_validation_data_configuration_list") übergebenen Liste der DataIdentifier und Datensatznummern.

[I: F-LAH_RxSWIN-371]

M - In der Diag_Status-Botschaft des Diagnostic-Server wird im Signal [Ident_valid] der Wert '1' angezeigt, dass die Identifikationsdaten vollständig gültig sind, d. h. der Hashwert wurde aktualisiert.

[I: F-LAH_RxSWIN-527]

*Nur relevant für SFD-E2E abgesicherte Diagnose-Systeme. Nicht relevant für unverriegelte Diagnose-Systeme mit Gruppenfreischaltung in der Produktion, vor ZP8.

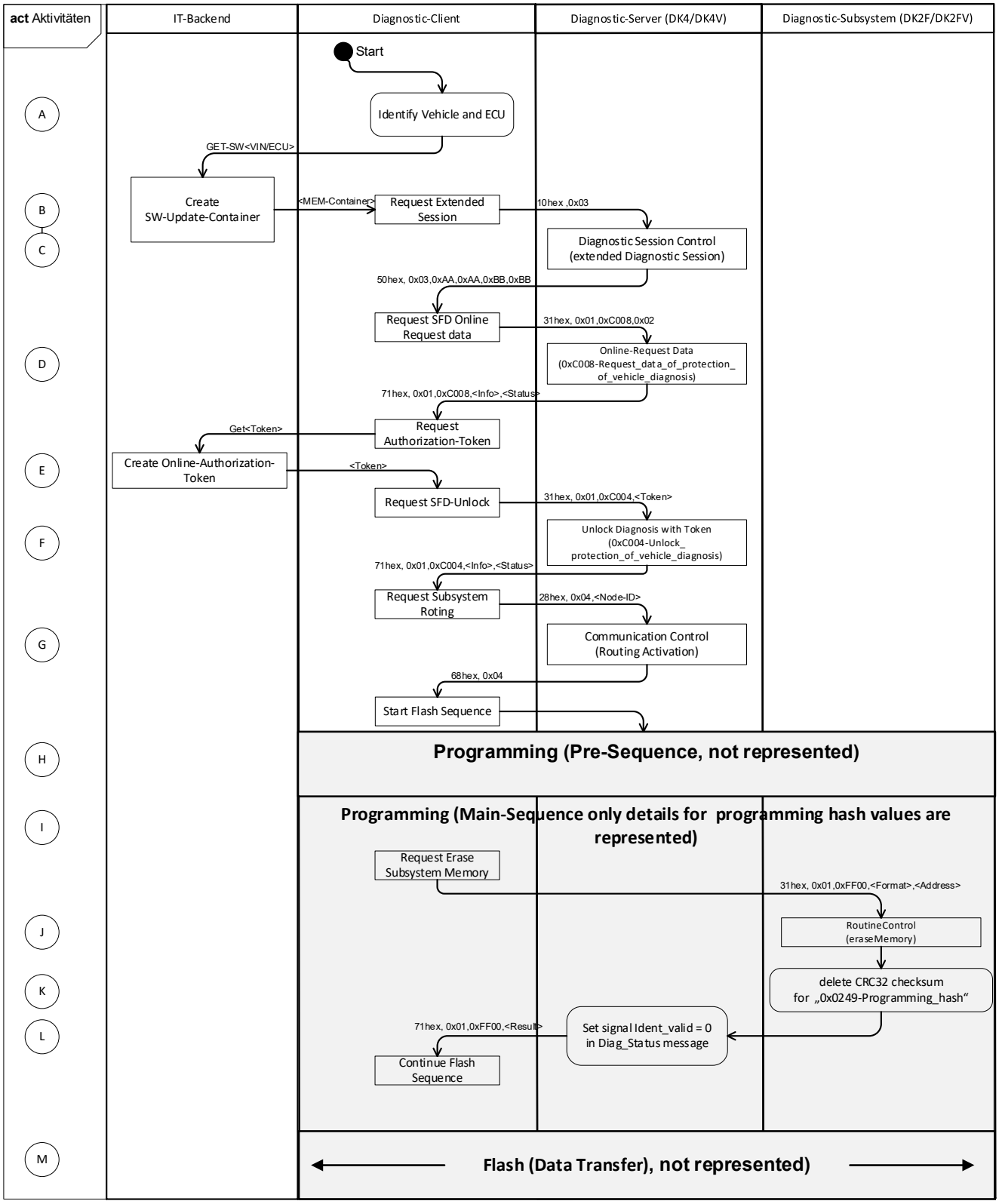
4.6.3 Beispielhafte Programmierung eines DK2F/DK2FV-Systems

[I: F-LAH_RxSWIN-303]

Ablauf eines Software-Updates eines DK2F/DK2FV System über ein DK4/DK4V-System durch einen Diagnose-Client.

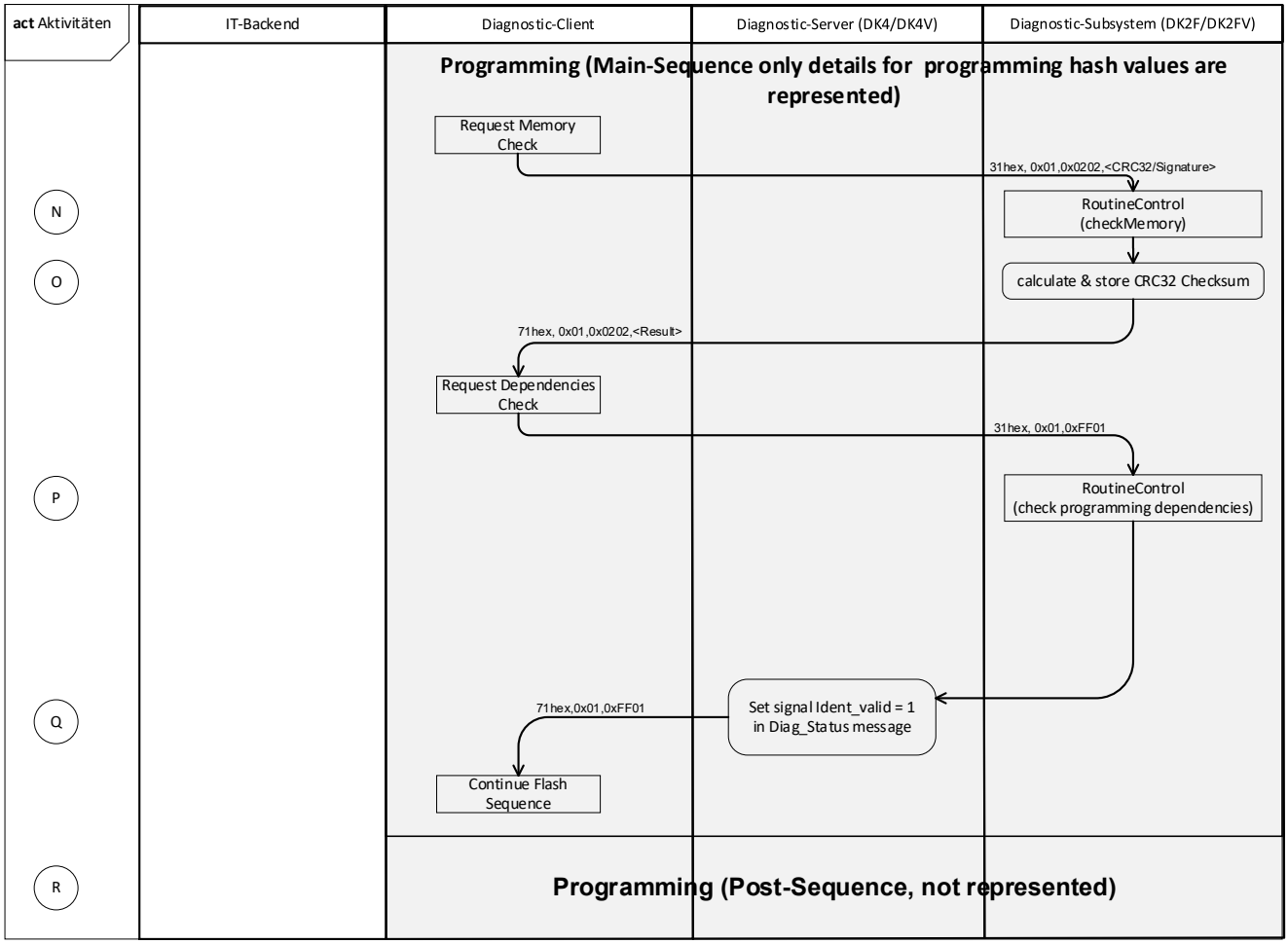
[I: F-LAH_RxSWIN-305]

Abbildung 4-15 Beispielhafte Programmierung DK2F/DK2FV-Systems, Teil 1/2



[I: F-LAH_RxSWIN-306]

Abbildung 4-16 Beispielhafte Programmierung eines DK2F/DK2FV-Systems, Teil 2/2



[I: F-LAH_RxSWIN-307]

A - Der Diagnostic-Client identifiziert das aktuelle Fahrzeug anhand der Fahrgestellnummer und das zu aktualisierende Steuergerät anhand seiner Identifikationsdaten.

[I: F-LAH_RxSWIN-310]

B - Der Diagnostic-Client lädt das entsprechende Software Update Paket (MEM-Container), welches durch das IT-Backend zusammengestellt wurde, herunter.

[I: F-LAH_RxSWIN-311]

C - Der Diagnostic-Client fordert für den Diagnostic-Server den Sessionwechsel in die Extended-Diagnostic-Session 0x03 an.

[I: F-LAH_RxSWIN-312]

D - Der Diagnostic-Client fordert von dem Diagnostic-Server die Anfragedaten für eine SFD Online Freischaltung an.

[I: F-LAH_RxSWIN-313]

E - Der Diagnostic-Client fragt am SFD-Backend mit den Anfragedaten ein Freischalt-Token an.

[I: F-LAH_RxSWIN-314]

F - Der Diagnostic-Client schaltet den Diagnostic-Server mit dem Freischalt-Token frei, um das Routing im Diagnostic-Server über die Diagnose aktivieren zu können.

[I: F-LAH_RxSWIN-315]

G - Der Diagnostic-Client aktiviert in dem Diagnostic-Server das Routing der Diagnose-Botschaften zu dem Diagnostic-Subsystem.

[I: F-LAH_RxSWIN-316]

H – Programmierablauf Pre-Sequenz gemäß Dokument /6/.

[I: F-LAH_RxSWIN-317]

I – Programmierablauf Main-Sequenz gemäß Dokument /6/.

[I: F-LAH_RxSWIN-318]

J - Der Diagnostic-Client fordert das Löschen eines logischen Blocks des Diagnostic-Subsystem an.

[I: F-LAH_RxSWIN-319]

K - Das Löschen des Speichers über die Routine „0xFF00-Erase Memory“ bewirkt, dass gleichzeitig die CRC32-Checksumme für den zu löschenden Block gelöscht wird.

[I: F-LAH_RxSWIN-523]

L - Das DK4/DK4V-System signalisiert über die Diag_Status Botschaft im Signal „Ident_valid“ = 0, dass die Steuergeräte Identifikation nicht mehr gültig ist.

[I: F-LAH_RxSWIN-320]

M - Der Diagnostic-Client führt die Flash-Sequenz mit dem Diagnostic-Subsystem durch.

[I: F-LAH_RxSWIN-321]

N - Der Diagnostic-Client fordert das Diagnostic-Subsystem nach der Flash-Sequenz auf, den korrekten Empfang des Speicherblocks zu überprüfen.

[I: F-LAH_RxSWIN-322]

O - Der Diagnostic-Server routet den Request zur Überprüfung des Speichers an das Diagnostic-Subsystem direkt weiter. Das Diagnostic-Subsystem berechnet die CRC32-Checksumme des logischen Bockes und speichert sie persistent ab.

[I: F-LAH_RxSWIN-323]

P - Der Diagnostic-Client fordert das Diagnostic-Subsystem nach der Flash-Sequenz auf, den korrekt empfangenen Speicherblock hinsichtlich Kompatibilität/Plausibilität zu überprüfen.

[I: F-LAH_RxSWIN-525]

Q - Der Diagnostic-Server signalisiert über die Diag_Status Botschaft das die Steuergeräte Identifikation gültig ist.

[I: F-LAH_RxSWIN-324]

R - Programmierablauf Post-Sequenz gemäß Dokument /6/.

5 Diagnoseobjekte

[I: F-LAH_RxSWIN-54]

Die folgende Tabelle zeigt die Übersicht der SUMS-relevanten Diagnosefunktionen bezogen auf die Diagnoseklasse.

[allg. Anf.: F-LAH_RxSWIN-180]

Tabelle 5-1 Diagnosefunktion und Diagnoseklassen

	DK4-high	DK4-low	DK4V-low	DK4*)	DK3	DK3V	DK2	DK2F	DK2FV	SWCL	Bemerkung
Diagnoseobjekt/-funktion											
0xF18F-Regulation_x_software_ identification_numbers	x	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	Nur einmal pro Fahrzeug im zentralen Gateway umzusetzen
SW-Teilnummern im DID "0xF187-VW Spare Part Number"	x	x	x	x	x	x	x	x	x	x	
SW-Version im DID "0xF189-VW Softw are Version Number"	x	x	x	x	x	x	x	x	x	x	
HW-Teilnummer im DID "0xF191-VW ECU Hardware Number"	x	x	NoImp	NoImp	x	NoImp	x	x	NoImp	NoImp	
HW-Version im DID "0xF1A3-VW ECU Hardware Version Number"	x	x	NoImp	NoImp	x	NoImp	x	x	NoImp	NoImp	
Fahrgestellnummer im DID "0xF190-Vehicle Identification Number"	(x)	(x)	(x)	(x)	(x)	(x)	NoImp	NoImp	NoImp	NoImp	VIN nur bei WFS-Teilnehmer, DoIP-Server und zGW
Konfigurations-Hashwert im DID "0245-Configuration_hash"	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	x	x	x	NoImp	Hashwert der Konfiguration
Sammel-DID 0x0248-Slave_list_ configuration_hash für "0x0245-Configuration_hash"	NoImp	x	x	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	Bei DK4-high nur mit unterlagerten DK2-Systemen
Programmierungs-Hashwert im DID "0249-Programming_hash"	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	x	x	NoImp	Hashwert des Programms
Sammel-DID 0x0247-Slave_list_programming_ hash für "0x0249-Programming_hash"	NoImp	x	x	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	Bei DK4-high nur mit unterlagerten DK2-Systemen
0x0251_Write_generic_to_sub_system	NoImp	x	x	x	NoImp	NoImp	NoImp	NoImp	NoImp	NoImp	Bei DK4-high nur mit unterlagerten DK2-Systemen
Liste der relevanten DIDs für die Hash-Berechnung im DID "0x0250-List_of_ configuration_data_identifier"	x	x	x	x	x	x	x	x	x	x	Bei DK2/DK2F: Bedatung erfolgt über den Busmaster/Host mit 0x0251
Berechnung der Integrity Validation Data mit RID "0x0253-Calculate_integrity_validation_data"	x	x	x	x	x	x	NoImp	NoImp	NoImp	x	Hashwert der Programm und Daten
Berechnung der Einzel-Hashwerte mit RID "0x0254-Calculate_individual_hash_value"	x	x	x	x	x	x	NoImp	NoImp	NoImp	x	Einzel-Hashwert der Konfigurations-daten
Programming-Hashwert für Q-LAH 80127 v4.0-Subsysteme (DID-Bereich 0xA800 - 0xA9FF)	NoImp	(x)	(x)	x	NoImp	NoImp	NoImp	x	NoImp	NoImp	Nur für DK4 mit DK2F in Bestands-plattformen (individueller DID für 0608er-Identifikation nach 80127 bis v4.0)
Konfigurations-Hashwert für Q-LAH 80127 v4.0-Subsysteme (DID-Bereich 0xAA00 - 0xABFF)	NoImp	(x)	(x)	x	NoImp	NoImp	x	x	NoImp	NoImp	

[I: F-LAH_RxSWIN-855]

*) = DK4 entsprechend Q-LAH 80127 bis v4.0 (ohne Unterteilung in DK4-low/DK4-high)

[I: F-LAH_RxSWIN-856]

x = beinhaltet

[I: F-LAH_RxSWIN-857]

(x) = optional

5.1 DataIdentifier

[allg. Anf.: F-LAH_RxSWIN-933]

Tabelle 5-2 Übersicht der Eigenschaften der DataIdentifier

			DiagnosticSession		
			Default Session	NonDefaultSession	
				ECUProgram-mingSession	Extended-Diagnostic-Session
Diagnoseobjekte (DID)	**SID**	**Security Level**	**0x01**	**0x02**	**0x03**
0xF1A3-VW_ECU_Hardware_version_number	22hex	NoImp	Available	Available	Available
	2Ehex	NoImp	notAvailable	notAvailable	notAvailable
0xF1B8-VW_system_firmware_versions	22hex	NoImp	NoImp	NoImp	NoImp
	2Ehex	NoImp	NoImp	NoImp	NoImp
0xF1A0-VW_data_set_number_or_ECU_data_container_number	22hex	NoImp	NoImp	NoImp	NoImp
	2Ehex	NoImp	NoImp	NoImp	NoImp
0xF1A1-VW_data_set_version_number	22hex	NoImp	NoImp	NoImp	NoImp
	2Ehex	NoImp	NoImp	NoImp	NoImp
0x0249-Programming_hash	22hex	NoImp	Available	notAvailable	Available
	2Ehex	NoImp	notAvailable	notAvailable	notAvailable
0x0247-Slave_list_programming_hash	22hex	NoImp	Available	notAvailable	Available
	2Ehex	NoImp	notAvailable	notAvailable	notAvailable
0x0245-Configuration_hash	22hex	NoImp	Available	notAvailable	Available
	2Ehex	NoImp	notAvailable	notAvailable	notAvailable
0x0248-Slave_list_configuration_hash	22hex	NoImp	Available	notAvailable	Available
	2Ehex	NoImp	notAvaillable	notAvailable	notAvailable
0xF18F-Regulation_x_software_identification_number	22hex	NoImp	Available	notAvailable	Available
	2Ehex	Old platform: SFD-BASIC New platform: SFD-E2E	notAvailable	notAvailable	Available
0x0250-Integrity_validation_data_configuration_list	22hex	NoImp	Available	notAvailable	Available
	2Ehex	SFD-E2E	notAvailable	notAvailable	C1
0x0251-Write_generic_to_sub_system	22hex	NoImp	notAvailable	notAvailable	notAvailable
	2Ehex	Old platform: C2 New platform: SFD-E2E	notAvailable	notAvailable	Available

[I: F-LAH_RxSWIN-935]

C1- Bei Systemen, die weder über Anpassung/Codierung noch über Datensätze in der Applikation konfigurierbar sind, muss dieser DataIdentifier nicht schreibbar sein, da in diesem Fall keine SFDE2E-Absicherung für die Konfiguration erforderlich ist. In diesem Fall müssen alle möglichen Bootloader-Datensätze in der Response des DataIdentifier "0x0250-Integrity_validation_data_configuration_list" enthalten sein.

[I: F-LAH_RxSWIN-936]

C2 - Das Zugriffsschutzverfahren (sofern vorhanden) des DK4-low ist zu verwenden.

5.1.1 0xF1A3-VW ECU Hardware Version Number

[I: F-LAH_RxSWIN-664]

Dieser DataIdentifier kennzeichnet die Hardwareversion eines Steuergerätes.

[allg. Anf.: F-LAH_RxSWIN-701]

Folgende Änderungen gegenüber Q-LAH 80125 bis Version 5.8 sind zu beachten:

[allg. Anf.: F-LAH_RxSWIN-806]

Entfall von:

- Die Hardwareversion ist nicht steuerungsrelevant für den Fahrzeugproduktionsprozess oder den Ersatzteilhandel.
- Ob eine software- und hardwarekompatible Änderung durch die Hardwareversion dokumentiert wird liegt im Ermessen des BSB in Absprache mit dem Auftragnehmer.

[allg. Anf.: F-LAH_RxSWIN-805]

Neu:
Folgende Änderung sind durch die Hardwareversion zu dokumentieren:

- Änderung von aktiven Bauteilen (z. B. µC, µP, RAM, Flash, ASIC)
- Teileabkündigung von passiven Bauteilen
- Änderungen an der Platine z. B. bzgl. EMV, Stromaufnahme
- Änderungen am funktionalen Verhalten: Z. B. optionaler Vorhalt wird bestückt
- Änderung beeinflusst Schnittstellen

5.1.2 0xF1A0-VW Data Set Number Or ECU Data Container Number

[allg. Anf.: F-LAH_RxSWIN-765]

Folgende Änderung gegenüber Q-LAH 80125 bis Version 5.8 ist zu beachten:

- Die Umsetzung das DataIdentifier "0xF1A0-VW Data Set Number Or ECU Data Container Number" im Diagnose-Server ist nicht mehr zulässig.

[Prozess-Anf.: F-LAH_RxSWIN-802]

Die Dokumentation der per ZDC geschriebenen Konfigurationsdaten erfolgt durch den Konfigurations-Hashwert.

[Prozess-Anf.: F-LAH_RxSWIN-766]

Hinweis: Die Teilnummer des Zieldatencontainers muss weiterhin in der Bauzustands-Dokumentation (Kapitel "RxSWIN-spezifische Dokumentation") enthalten sein.

5.1.3 0xF1A1-VW Data Set Version Number

[allg. Anf.: F-LAH_RxSWIN-799]

Folgende Änderung gegenüber Q-LAH 80125 bis Version 5.8 ist zu beachten:

- Die Umsetzung des DataIdentifier "0xF1A1-VW Data Set Version Number" im Diagnose-Server ist nicht mehr zulässig.

[Prozess-Anf.: F-LAH_RxSWIN-800]

Die Dokumentation der per ZDC geschriebenen Konfigurationsdaten erfolgt durch den Konfigurations-Hashwert.

[Prozess-Anf.: F-LAH_RxSWIN-801]

Hinweis: Die Version des Zieldatencontainers muss weiterhin in der Bauzustands-Dokumentation (Kapitel "RxSWIN-spezifische Dokumentation") enthalten sein.

5.1.4 0x0249-Programming_hash

[allg. Anf.: F-LAH_RxSWIN-449]

Tabelle 5-3 Aufbau DataRecord DataIdentifier 0x0249-Programming_hash

DID	**0x0249**
Bezeichnung	Programming_hash
Beschreibung	Dieser DataIdentifier beinhaltet die Integrity Validation Data der Programmdaten. Die Berechnung des Hashwertes erfolgt über die gesamte Software.
Convention	alle Server
Diagnoseklasse	DK2F/DK2FV
Session	APP: 0x01 (R), 0x03 (R) BLF: NoImp
SecurityLevel	NoImp
Changing	APP
Format	<[Programming_hash] 32-Byte-Hex>
Range	[Programming_hash] 00..00hex - FF..FFhex
Init	entfällt, immer verfügbar
Beispiel	-
Datenkategorie	Analysedaten
eBZD	M

5.1.5 0x0247-Slave_list_programming_hash

[allg. Anf.: F-LAH_RxSWIN-443]

Tabelle 5-4 Aufbau DataRecord DataIdentifier 0x0247-Slave_list_programming_hash

DID	**0x0247**
Bezeichnung	Slave_list_programming_hash
Beschreibung	Dieser DataIdentifier beinhaltet alle Integrity Validation Data über Instructioncode der unterlagerten DK2F/DK2FV-Systeme. Es gelten folgende Festlegungen: • Der jeweilige Inhalt entspricht dem Parameter [Programming_hash] des DataIdentifier "0x0249-Programming_hash" • Dem jeweiligen Inhalt muss die jeweilige SubSystemNodeAddress vorangestellt werden.
Convention	Busmaster
Diagnoseklasse	DK4-low/DK4V-low
Session	APP: 0x01 (R), 0x03 (R) BLF: NoImp
SecurityLevel	NoImp
Changing	APP
Format	<[NumberOfExpectedSubSystemIdentification] 1-Byte-Hex>+ <[NumberOfRetrievedSubSystemIdentification] 1-Byte-Hex>+ <n-mal<[SubSystemNodeAddress] 2-Byte-Hex>+ <[Programming_hash] 32-Byte-Hex>> (n: Anzahl der DK2F/DK2F-Systeme)
Range	[NumberOfExpectedSubSystemIdentification]: 00hex - FFhex [NumberOfRetrievedSubSystemIdentification]: 00hex - FFhex [SubSystemNodeAddress]: 00 00hex - FF FFhex [Programming_hash]: siehe DID 0x0249
Init	entfällt, immer verfügbar
Beispiel	-
Datenkategorie	Analysedaten
eBZD	NoImp

5.1.6 0x0245-Configuration_hash

[allg. Anf.: F-LAH_RxSWIN-445]

Tabelle 5-5 Aufbau DataRecord DataIdentifier 0x0245-Configuration_hash

DID	**0x0245**
Bezeichnung	Configuration_hash
Beschreibung	Dieser DataIdentifier beinhaltet die Integrity Validation Data der Konfigurationsdaten.
Convention	alle Server
Diagnoseklasse	DK2/DK2F/DK2FV
Session	APP: 0x01 (R), 0x03 (R) BLF: NoImp
SecurityLevel	NoImp
Changing	APP
Format	<[Configuration_hash] 32-Byte-Hex>
Range	[Configuration_hash] 00..00hex - FF..FFhex
Init	entfällt, immer verfügbar
Beispiel	-
Datenkategorie	Analysedaten
eBZD	M

5.1.7 0x0248-Slave_list_configuration_hash

[allg. Anf.: F-LAH_RxSWIN-447]

Tabelle 5-6 Aufbau DataRecord DataIdentifier 0x0248-Slave_list_configuration_hash

DID	**0x0248**
Bezeichnung	Slave_list_configuration_hash
Beschreibung	Dieser DataIdentifier beinhaltet alle Integrity Validation Data über Konfigurationsdaten der unterlagerten DK2/DK2F/DK2FV-Systeme. Es gelten folgende Festlegungen: • Der jeweilige Inhalt entspricht dem Parameter [Configuration_hash] des DataIdentifier "0x0245-Configuration_hash". • Dem jeweiligen Inhalt muss die jeweilige SubSystemNodeAddress vorangestellt werden.
Convention	Busmaster
Diagnoseklasse	DK4-low/DK4V-low
Session	APP: 0x01 (R), 0x03 (R) BLF: NoImp
SecurityLevel	NoImp
Changing	APP
Format	<[NumberOfExpectedSubSystemIdentification] 1-Byte-Hex>+ <[NumberOfRetrievedSubSystemIdentification] 1-Byte-Hex>+ <n-mal<[SubSystemNodeAddress] 2-Byte-Hex>+ <[Configuration_hash] 32-Byte-Hex>> (n: Anzahl der DK2/DK2F/DK2F-Systeme)
Range	[NumberOfExpectedSubSystemIdentification]: 00hex - FFhex [NumberOfRetrievedSubSystemIdentification]: 00hex - FFhex [SubSystemNodeAddress]: 00 00hex - FF FFhex [Configuration_hash]: siehe DID 0x0245
Init	entfällt, immer verfügbar
Beispiel	-
Datenkategorie	Analysedaten
eBZD	NoImp

5.1.8 0xF18F-Regulation_x_software_identification_numbers

[allg. Anf.: F-LAH_RxSWIN-14]

Tabelle 5-7 Aufbau DataRecord DataIdentifier 0xF18F-Regulation_x_software_identification_numbers

DID	**0xF18F**
Bezeichnung	Regulation_x_software_identification_numbers
Beschreibung	Dieser DataIdentifier beinhaltet die Liste der Software-Identifikationsnummern von z.B. UNECE- oder in chinesischen Normen definierten GB/T-Funktionen, die in einem Fahrzeug verfügbar sind. Diese Liste ist fahrzeugspezifisch. Sämtliche Nutzinhalte wie [Length_of_RxSWIN], [Regulation_identification], [Separation_character] und [Software_identification] dürfen vom Diagnose-Server nicht plausibilisiert werden und müssen unverändert ausgegeben werden.
Convention	Steuergerät mit zentralem Diagnosezugang
Diagnoseklasse	DK4-high
Session	APP: 0x01 (R), 0x03 (R/W) BLF: NoImp
SecurityLevel	Steuergeräte in Bestandsplattformen: SFD-Zugriffsschutz, Rolle BASIC (W) Steuergerät in neuen Plattformen: SFD-E2E (W)
Changing	DIAG
Format	<n-mal <[Length_of_RxSWIN] 1-Byte-Hex>+ <[Regulation_identification] m-Byte-ASCII, variabel>+ <[Separation_character] 1-Byte-ASCII>+ <[Software_identification] 9..11-Byte-ASCII, variabel> > (n: Anzahl der RxSWINs, variabel; m: Anzahl der Bytes für die Regulierungs-ID, variabel)
Range	[Length_of_RxSWIN] 00-FFhex [Regulation_identification] 21-7Ahex [Separation_character] 20hex (ASCII "SPACE") [Software_identification] 30-39hex, 41-5Ahex, 61-7A Andere Wert sind ISO-reserved
Init	2D 2D 2D 2D 2Dhex (' - - - - - ')
Beispiel	0F 52 30 37 39 20 76 30 35 37 34 31 37 35 33 61 13 47 42 2F 54 33 36 30 34 37 20 76 30 34 33 36 39 38 35 32 Anzahl RxSWIN = 2 RxSWIN #1: Length_of_RxSWIN: 0x0F = 15 Byte Regulation_identification: R079 Software_identification: v05741753a RxSWIN #2: Length_of_RxSWIN: 0x13 = 19 Byte Regulation_identification: GB/T36047 Software_identification: v04369852
Datenkategorie	Prozessparameter
eBZD	M

[I: F-LAH_RxSWIN-70]

Die Gesamtlänge der RxSWIN soll mindestens 50 Regulierungen und die dazugehörigen regulierungsbezogenen Software-Identifikationen beinhalten können.

[allg. Anf.: F-LAH_RxSWIN-71]

Es ergibt sich daraus eine Mindestgröße des Speichers für den DataIdentifier "0xF18F-Regulation_x_software_identification_numbers" von 1000 Byte.

[allg. Anf.: F-LAH_RxSWIN-67]

Das Lesen des DataIdentifiers "0xF18F-Regulation_x_software_identification_numbers" über den Service ReadDataByIdentifier (22hex) darf nicht über ein Zugriffsschutzverfahren geschützt werden.

[allg. Anf.: F-LAH_RxSWIN-66]

Für Steuergeräte in Bestandsplattformen gilt: Das Schreiben des DataIdentifier "0xF18F-Regulation_x_software_identification_numbers" über den Service WriteDataByIdentifier (2Ehex) ist über "SFD-Zugriffsschutz (Rolle BASIC)" entsprechend Dokument /4/ abzusichern.

[allg. Anf.: F-LAH_RxSWIN-340]

Für Steuergeräte in neuen Plattformen gilt: Das Schreiben des DataIdentifier "0xF18F-Regulation_x_software_identification_numbers" über den Service WriteDataByIdentifier (2Ehex) ist über "SFD-Ende-zu-Ende-Absicherung (SFD-E2E)" entsprechend Dokument /4/ abzusichern.

5.1.8.1 Anforderungen an Prozesse

[Prozess-Anf.: F-LAH_RxSWIN-72]

Für nicht im Bauauftrag des Fahrzeuges vorhandene UNECE-Funktionen wird keine RxSWIN geschrieben. D. h. für nicht in dem konkreten Fahrzeug vorhandene UNECE-Funktionen darf es keinen Listeneintrag im DataIdentifier "0xF18F-Regulation_x_software_identification_numbers" mit einer RxSWIN geben.

[Prozess-Anf.: F-LAH_RxSWIN-74]

Der DataIdentifier "0xF18F-Regulation_x_software_identification_numbers" besitzt eine fahrzeugspezifische Länge. Der Bedatungs-Prozess muss dafür sorgen, dass der DataIdentifier "0xF18F-Regulation_x_software_identification_numbers" entsprechend der Anforderung F-LAH_RxSWIN-72 berechnet wird.

[I: F-LAH_RxSWIN-555]

Hinweis:
Die Umsetzung kann z. B. durch einen JAVA- oder OTX-Job erfolgen.

5.1.8.2 Anforderungen an IT-Systeme

[Prozess-Anf.: F-LAH_RxSWIN-220]

Die Zusteuerung des Datencontainers mit der Liste der RxSWINs erfolgt in der Datenlogistik über ein, dem Gateway-Steuergerät zugeordneten, DK2V-System.

[Prozess-Anf.: F-LAH_RxSWIN-803]

Hinweis: Hierbei handelt es sich nicht um ein DK2V-System im Sinne der Dokumente /3/ bzw. /12/. Dieses DK2V-System beinhaltet lediglich eine Diagnoseadresse, welche zur Steuerung der Daten in der Logistikkette notwendig ist.

5.1.9 0x0250-Integrity_validation_data_configuration_list

[allg. Anf.: F-LAH_RxSWIN-349]

Tabelle 5-8 Aufbau DataRecord DataIdentifier 0x0250-Integrity_validation_data_configuration_list

DID	**0x0250**
Bezeichnung	Integrity_validation_data_configuration_list
Beschreibung	Dieser DataIdentifier beinhaltet die Liste der Identifier (DataIdentifier und Datensatznummern, die für die Hashwert-Berechnung von Konfigurationsdaten verwendet werden. Dieser DataIdentifier wird initial mit der Liste der DataIdentifier und den Datensatznummern, die im BT-LAH als ZDC-relevant gekennzeichnet sind, gefüllt. Im Rahmen des ZDC-Prozess kann diese Liste verändert werden.
Convention	ZDC
Diagnoseklasse	DK2/DK2F/DK2FV, DK3/DK3V, DK4/DK4V, SWCL
Session	APP: 0x01 (R), 0x03 (R/W*) BLF: NoImp
SecurityLevel	SFD-E2E (W)
Changing	DIAG
Format	<<[Number_of_identifier] 2-Byte-Hex>+ n-mal<[Identifier] 2-Byte-Hex>> (n: Anzahl der Identifier)
Range	[Number_of_identifier] 00 00hex = Kein DataIdentifier oder keine Datensatznummer vorhanden 00 01hex - FF FFhex [Identifier] 00 00-FF FFhex
Init	Initialisierung mit den DataIdentifier und den Datensatznummer, die im BT-LAH, als ZDC-relevant gekennzeichnet sind.
Beispiel	00 04 12 43 98 67 FE CD 71 01 Anzahl Identifier = 4 DID#1: 0x1243 DID#2: 0x9867 DID#3: 0xFECD DSNo: 0x7101
Datenkategorie	Fahrzeugparameter
eBZD	M

[allg. Anf.: F-LAH_RxSWIN-837]

*) Bei Systemen, die weder über Anpassung/Codierung noch über Datensätze in der Applikation konfigurierbar sind, muss dieser DataIdentifier nicht schreibbar sein, da in diesem Fall keine SFD-E2E-Absicherung für die Konfiguration erforderlich ist. In diesem Fall müssen alle möglichen Bootloader-Datensätze in der Response des DataIdentifier "0x0250-Integrity_validation_data_configuration_list" enthalten sein.

[allg. Anf.: F-LAH_RxSWIN-539]

Der Inhalt des DataIdentifier "0x0250-Integrity_validation_data_configuration_list" muss persistent im Steuergerät gespeichert werden.

[allg. Anf.: F-LAH_RxSWIN-551]

Weicht die in dem DID "0x0250-Integrity_validation_data_configuration_list" initial verwendete Liste der DataIdentifier und Datensatznummern, von der aus dem Bauauftrag resultierenden Einträgen in der Liste ab, dann muss im Rahmen des ZDC-Prozesses der DID "0x0250-Integrity_validation_data_configuration_list" entsprechend konfiguriert werden.

5.1.10 0x0251-Write_generic_to_sub_system (Schreiben des DataIdentifiers 0x0250 über das DK4-low/DK4V-low-System zum unterlagerten DK2/DK2F/DK2FV-System)

[I: F-LAH_RxSWIN-494]

Das Schreiben des DataIdentifier "0x0250-Integrity_validation_data_configuration_list" erfolgt über den Service WriteDataByIdentifier (2Ehex) und dem DataIdentifier "0x0251-Write_generic_to_sub_system" über das DK4-Low-System. Zur Adressierung des unterlagerten Zielsystems wird im 2Ehex-Request im DataRecord an den ersten beiden Bytes die SubSystemNodeAddress (SSN) verwendet. Die folgenden beiden Bytes beinhalten den DataIdentifier 0x0250 gefolgt von den Nutzdaten.
Hinweis: Dies ist eine Änderung gegenüber Q-LAH 80124 bis Version 2.8 (Q-LAH_80124-7984).

[allg. Anf.: F-LAH_RxSWIN-615]

Tabelle 5-9 Aufbau DataRecord DataIdentifier 0x0251-Write_generic_to_sub_system

DID	**0x0251**
Bezeichnung	Write_generic_to_sub_system
Beschreibung	Dieser DataIdentifier wird zum Schreiben von Daten für Subsysteme über ein DK4-low-System verwendet. Das Auslesen von Daten ist mittels dieses DataIdentifiers nicht möglich. (read only)
Convention	Busmaster
Diagnoseklasse	DK4/DK4-low
Session	APP: 0x03 (W) BLF: NoImp
SecurityLevel	Steuergeräte in Bestandsplattformen: Falls vorhanden ist das bestehende Zugriffsschutzverfahren des DK4-low zu verwenden. Steuergerät in neuen Plattformen: SFD-E2E (W)
Changing	DIAG
Format	<<[Target_sub_system_node_address] 2-Byte-Hex>+ <[Target_data_identifier] 2-Byte-Hex>+ <n-mal [Data_record_target_data_identifier] 1-Byte-Hex>> (n: 1..256, variabel)
Range	[Target_sub_system_node_address] 00 00 - FF FFhex [Target_data_identifier] 0x02 50 [Data_record_target_data_identifier] 00 - FFhex Andere Wert sind VOLKSWAGEN AG-reserved
Init	kein Init im Server
Beispiel	01 41 02 50 13 24 97 86 Target_sub_system_node_address: 0x0141 = Kartenleser TV Tuner Target_data_identifier: 0x02 50 Data_record_target_data_identifier: 0x13 24 97 86
Datenkategorie	Prozessparameter
eBZD	NoImp

5.1.10.1 Request Message Definition

[I: F-LAH_RxSWIN-500]

Folgende Parameter sind zu implementieren:

[allg. Anf.: F-LAH_RxSWIN-495]

Tabelle 5-10 Request Message Definition

Data	Description		Cvt.	Value (Hex)
#1	Request SID	WriteDataByIdentifier	M	2E
#2	DataIdentifier#1	Write_generic_to_sub_system [Byte#1] MSB	M	02
#3	DataIdentifier#2	Write_generic_to_sub_system [Byte#2]	M	51
#4	DataRecord#1	Target_sub_system_node_address [Byte#1] MSB	M	00-FF
#5	DataRecord#2	Target_sub_system_node_address [Byte#2]	M	00-FF
#6	DataRecord#3	Target_data_identifier [Byte#1] MSB	M	00-FF
#7	DataRecord#4	Target_data_identifier [Byte#2]	M	00-FF
#8	DataRecord#5	Data_record_target_data_identifier#1	M	00-FF
:	:		:	:
#9+m-1	DataRecord#5+m-1	Data_record_target_data_identifier#m	U	00-FF

5.1.10.2 Request Message Parameter Definition

[I: F-LAH_RxSWIN-497]

Folgende Parameter sind zu implementieren:

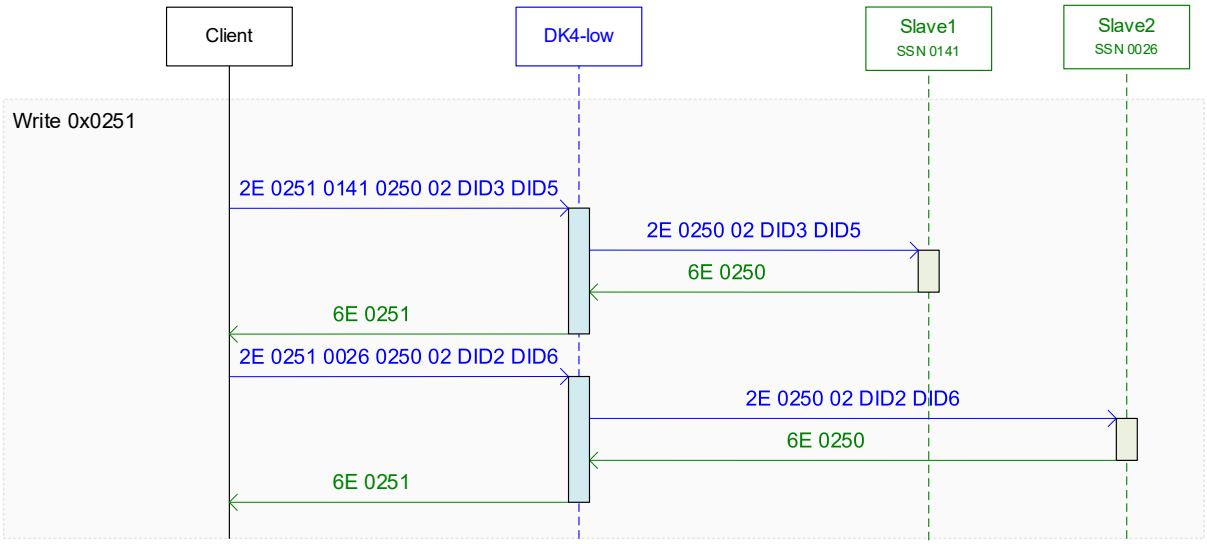
[allg. Anf.: F-LAH_RxSWIN-498]

Tabelle 5-11 Request Message Parameter Definition

Definition
Target_sub_system_node_address Dieser Parameter gibt die SubSystemNodeAddress (SSN) des unterlagerten Zielsystems an.
Target_data_identifier Dieser Parameter gibt den Wert des DataIdentifiers an. Erlaubter Wert ist hier 0x0250. Andere Werte sind VOLKSWAGEN AG-reserved
Data_record_target_data_identifier Dieser Parameter gibt den DataRecord des Parameters [Target_data_identifier] an.

[allg. Anf.: F-LAH_RxSWIN-475]

Abbildung 5-1 Schreiben von 0x0250-Integrity_validation_data_configuration_list



5.2 RoutineIdentifier

[allg. Anf.: F-LAH_RxSWIN-934]

Tabelle 5-12 Übersicht der Eigenschaften der RoutineIdentifier

			DiagnosticSession		
				NonDefaultSession	
			Default Session	ECUProgram-mingSession	Extended-Diagnostic-Session
Diagnoseobjekte (RID)	**SID**	**Security Level**	**0x01**	**0x02**	**0x03**
0x03E7-Reset_to_factory_setting	31hex	NoImp	NoImp	NoImp	NoImp
0x0253-Calculate_integrity_validation_data	31hex	NoImp	Available	notAvaillable	Available
0x0254-Calculate_individual_hash_value	31hex	NoImp	Available	notAvaillable	Available

5.2.1 0x03E7-Reset_to_factory_setting

[I: F-LAH_RxSWIN-927]

Dieser RoutineIdentifier wird nach Dokument /1/ verwendet um alle Codierungen, Fahrzeug-, Kunden-, Werkstatt-, Prozessparameter, Erstbedatungs- und Lernwerte zurückzusetzen. Ein Ausführen der Routine führt damit zu einer Änderung der IVD-relevanten Anteile des Konfigurations-Hashwertes wie Codierungen, Fahrzeugparameter und Erstbedatungswerte.

[allg. Anf.: F-LAH_RxSWIN-928]

Folgende Änderung gegenüber Q-LAH 80124 bis Version 5.8 ist zu beachten:

- Die Umsetzung des RoutineIdentifier "0x03E7-Reset_to_factory_setting" im Diagnose-Server ist nicht mehr zulässig.

5.2.2 0x0253-Calculate_integrity_validation_data

[I: F-LAH_RxSWIN-451]

Dieser RoutineIdentifier wird zur Berechnung der Integrity Validation Data von DK3/DK3V/DK4/DK4V/SWCL für Instruction-Code und Konfigurationsdaten verwendet.

[allg. Anf.: F-LAH_RxSWIN-473]

Dieser RoutineIdentifier muss mit dem hier gefordertem Request-/Response-Verhalten und den hier geforderten Parametern umgesetzt werden. Die Verwendung der Subfunctions
• StopRoutine (0x02)
• RequestRoutineResults (0x03)
ist für diesen RoutineIdentifier nicht zulässig. Er liefert mit der pos. Response auf den RoutineControlType 0x01 sein Ergebnis und wird automatisch beendet.

[allg. Anf.: F-LAH_RxSWIN-597]

Das hier beschriebene Responseverhalten entspricht dem Q-LAH 80124, Version 2.x. Steuergeräte in Bestandsprojekten, die eine Version des Q-LAH 80124 1.x umgesetzt haben, müssen für den RoutineIdentifier "0x0253-Calculate_integrity_validation_data" die hier geforderte Response umsetzen. Bestehende Routinen auf Basis Q-LAH 80124 Version 1.x müssen im Steuergerät nicht angepasst werden.

[allg. Anf.: F-LAH_RxSWIN-616]

Tabelle 5-13 Aufbau RoutineIdentifier 0x0253-Calculate_integrity_validation_data

RID	**0x0253**
Bezeichnung	Calculate_integrity_validation_data
Beschreibung	Dieser RoutineIdentifier wird zur Berechnung der Integrity Validation Data für Instruction-Code und Konfigurationsdaten verwendet.
Convention	Alle Server
Diagnoseklasse	DK3/DK3V, DK4/DK4-low/DK4V-low, DK4-high/DK4V-high, SWCL
Session	APP: 0x01, 0x03 BLF: NoImp
SecurityLevel	NoImp
Changing	APP
Format	siehe Request/Response-Definition
Range	siehe Request/Response-Definition
Init	kein Init im Server
Beispiel	siehe Request/Response-Definition
eBZD	NoImp

5.2.2.1 Request Message Definition

[I: F-LAH_RxSWIN-453]

Folgende Request Message ist zu implementieren:

[allg. Anf.: F-LAH_RxSWIN-454]

Tabelle 5-14 Request Message Definition

Data	Description		Cvt.	Value (Hex)
#1	Request SID	RoutineControl	M	31
#2	RoutineControlType	StartRoutine	M	01
#3	RoutineIdentifier [Byte#1]	Calculate_integrity_validation_data [Byte#1] (MSB)	M	02
#4	RoutineIdentifier [Byte#2]	Calculate_integrity_validation_data [Byte#2]	M	53
#5	RoutineControlOption [Byte#1]	Type_of_calculation	M	00-FF
#6	RoutineControlOption [Byte#2]	Type_of_hashvalue	M	00-FF

5.2.2.2 Request Message Parameter Definition

[I: F-LAH_RxSWIN-457]

Folgende Parameter sind zu implementieren:

[allg. Anf.: F-LAH_RxSWIN-458]

Tabelle 5-15 Request Message Parameter Definition

Definition
Type_of_calculation Dieser Parameter gibt den Umfang der Hashw ertberechnung an. 0x00 = Die Hashw ertberechnung erfolgt über die Konfigurationsdaten (Data). Die Informationen über die Identifier w erden dem DataIdentifier 0x0250 entnommen. 0x01 = Die Hashw ertberechnung erfolgt über die Programmierdaten von Applikations-und Bootloader-Softw are (Instruction-Code). Andere Werte sind VOLKSWAGEN AG-reserved
Type_of_hashvalue Dieser Parameter gibt das Verfahren der Hashw ertberechnung an. 0x01 = SHA-256 Andere Werte sind VOLKSWAGEN AG-reserved

5.2.2.3 Positive Response Message Definition

[I: F-LAH_RxSWIN-460]

Folgende Response Message ist zu implementieren:

[allg. Anf.: F-LAH_RxSWIN-461]

Tabelle 5-16 Positive Response Message Definition

Data	Description		Cvt.	Value (Hex)
#1	Response SID	RoutineControl	M	71
#2	RoutineControlType	StartRoutine	M	01
#3	RoutineIdentifier [Byte#1]	Calculate_integrity_validation_data [Byte#1] (MSB)	M	02
#4	RoutineIdentifier [Byte#2]	Calculate_integrity_validation_data [Byte#2]	M	53
#5	RoutineStatusRecord#1	Result_of_calculation	M	00-FF
#6	RoutineStatusRecord#2	Typ_of_hashvalue	M	00-FF
#7	RoutineStatusRecord#3	Hashvalue [Byte#1] (MSB)	C1	00-FF
:	:	:	:	:
#6+n	RoutineStatusRecord#2+n	Hashvalue [Byte#n]	C1	00-FF

[allg. Anf.: F-LAH_RxSWIN-462]

C1 = Länge abhängig vom Parameter [Type_of_calculation]. Bei 'SHA-256' beträgt die Länge 32 Bytes.

5.2.2.4 Positive Response Message Parameter Definition

[I: F-LAH_RxSWIN-464]

Folgende Parameter sind zu implementieren:

[allg. Anf.: F-LAH_RxSWIN-465]

Tabelle 5-17 Positive Response Message Parameter Definition

Definition
Result_of_calculation 0x00 = Calculation_successful Die Berechnung des Hashwertes wurde erfolgreich durchgeführt. 0x01 = Calculation_failed Die Berechnung des Hashwertes ist fehlgeschlagen. 0x02 = Calculation_Identifier_not_found Mindestens ein DataIdentifier oder eine Datensatznummer aus dem DataIdentifier 0x0250 ist im Diagnose-Server nicht vorhanden. 0x03 - 0xFF = VOLKSWAGEN AG-reserved
Type_of_hashvalue 0x00 = VOLKSWAGEN AG-Reserved 0x01 = SHA-256 Die Berechnung des Hashwertes erfolgt nach dem SHA-256-Verfahren. 0x02 - 0xFF = VOLKSWAGEN AG-reserved
Hashvalue n Byte <0x00 - 0xFF> Berechneter Hashwert, bei SHA-256 entspricht n = 32.

5.2.3 0x0254-Calculate_individual_hash_value

[I: F-LAH_RxSWIN-868]

Dieser RoutineIdentifier wird zum Berechnen und Auslesen der Einzel-Hashwerte von DK3/DK3V/DK4/DK4V-Systemen und SWCL für Konfigurationsdaten verwendet.

[allg. Anf.: F-LAH_RxSWIN-930]

Dieser RoutineIdentifier muss mit dem hier gefordertem Request-/Response-Verhalten und den hier geforderten Parametern umgesetzt werden. Die Verwendung der Subfunctions
• StopRoutine (0x02)
• RequestRoutineResults (0x03)
ist für diesen RoutineIdentifier nicht zulässig. Er liefert mit der pos. Response auf den RoutineControlType 0x01 sein Ergebnis und wird automatisch beendet.

[allg. Anf.: F-LAH_RxSWIN-929]

Das hier beschriebene Responseverhalten entspricht dem Q-LAH 80124, Version 2.x. Steuergeräte in Bestandsprojekten, die eine Version des Q-LAH 80124 1.x umgesetzt haben, müssen für den RoutineIdentifier 0x0254-Calculate_individual_hash_value" die hier geforderte Response umsetzen. Bestehende Routinen auf Basis Q-LAH 80124 Version 1.x müssen im Steuergerät nicht angepasst werden.

[allg. Anf.: F-LAH_RxSWIN-871]

Tabelle 5-18 Aufbau RoutineIdentifier 0x0254-Calculate_individual_hash_value

RID	**0x0254**
Bezeichnung	Calculate_individual_hash_value
Beschreibung	Dieser RoutineIdentifier wird zum Berechnen und Auslesen von Einzel-Hashwerte der Konfigurationsdaten verwendet.
Convention	Alle Server
Diagnoseklasse	DK3/DK3V, DK4/DK4-low/DK4V-low, DK4-high/DK4V-high, SWCL
Session	APP: 0x01, 0x03 BLF: NoImp
SecurityLevel	NoImp
Changing	APP
Format	siehe Request/Response-Definition
Range	siehe Request/Response-Definition
Init	kein Init im Server
Beispiel	siehe Request/Response-Definition
eBZD	NoImp

5.2.3.1 Request Message Definition

[I: F-LAH_RxSWIN-873]

Folgende Request Message ist zu implementieren:

[allg. Anf.: F-LAH_RxSWIN-874]

Tabelle 5-19 Request Message Definition

Data	Description		Cvt.	Value (Hex)
#1	Request SID	RoutineControl	M	31
#2	RoutineControlType	StartRoutine	M	01
#3	RoutineIdentifier [Byte#1]	Calculate_individual_hash_value [Byte#1] (MSB)	M	02
#4	RoutineIdentifier [Byte#2]	Calculate_individual_hash_value [Byte#2]	M	54
#5	RoutineControlOption [Byte#1]	Type_of_hashvalue	M	00-FF
#6	RoutineControlOption [Byte#2]	Type_of_hash	M	00-FF
#7	RoutineControlOption [Byte#3]	Individual_hash_value_id	M	00-FF
#8	RoutineControlOption [Byte#4]	Individual_hash_value_id	M	00-FF

5.2.3.2 Request Message Parameter Definition

[I: F-LAH_RxSWIN-876]

Folgende Parameter sind zu implementieren:

[allg. Anf.: F-LAH_RxSWIN-877]

Tabelle 5-20 Request Message Parameter Definition

Definition
Type_of_hashvalue 0x00 = VOLKSWAGEN AG-reserved 0x01 = SHA-256 Die Berechnung des Hashw ertes erfolgt nach dem SHA-256-Verfahren. 0x02 - 0xFF = VOLKSWAGEN AG-reserved
Type_of_hash 0x00 = Einzel-Hashw ert Datensatz (bei Datensatzdow nload Gen. 1) 0x01 = Einzel-Hashw ert Bootloader-Datensatz (bei Datensatzdow nload Gen. 2) 0x02 = Einzel-Hashw ert Applikations-Datensatz (entspricht der Applikations-Datensatz-Nummer Datensatzdow nload Gen. 2) 0x03 = Einzel-Hashw ert über alle Anpassungen/Codierungen gemäß DataIdentifier 0x0250 0x04 - 0xFE = VOLKSWAGEN AG-reserved 0xFF = alle Einzel-Hashw erte gemäß DataIdentifier 0x0250
Individual_hash_value_id Dieser Parameter identifiziert Einzel-Hashw erte von Applikationsdatensätzen gemäß DataIdentifier 0x0250. 0x7200 - 0x72FF = w ird nur verw endet bei [Type_of_hash] = 0x02 0x0000 = bei allen anderen Werte im Parameter [Type_of_hashvalue] Alle anderen Werte sind VOLKSWAGEN AG-reserved

5.2.3.3 Positive Response Message Definition

[I: F-LAH_RxSWIN-879]

Folgende Response Message ist zu implementieren:

[allg. Anf.: F-LAH_RxSWIN-880]

Tabelle 5-21 Positive Response Message Definition

Data	Description		Cvt.	Value (Hex)
#1	Response SID	RoutineControl	M	71
#2	RoutineControlType	StartRoutine	M	01
#3	RoutineIdentifier [Byte#1]	Calculate_individual_hash_value [Byte#1] (MSB)	M	02
#4	RoutineIdentifier [Byte#2]	Calculate_individual_hash_value [Byte#2]	M	54
#5	RoutineStatusRecord#1	State_of_hashvalue	M	00-FF
#6	RoutineStatusRecord#2	Typ_of_hashvalue	M	00-FF
#7	RoutineStatusRecord#3	Typ_of_hash	M	00-FF
#8	RoutineStatusRecord#4	Individual_hash_value_id [Byte#1] (MSB)	M	00-FF
#9	RoutineStatusRecord#5	Individual_hash_value_id [Byte#2]	M	00-FF
#10	RoutineStatusRecord#6	Hash_value#1 [Byte#1] (MSB)	C1	00-FF
:	:	:	:	:
#9+n	RoutineStatusRecord#6+n	Hash_value#1 [Byte#n]	C1	00-FF
#10 + n	RoutineStatusRecord#6+n+1	Hash_value#m [Byte#1] (MSB)	C2	00-FF
:	:	:	:	:
#9+n + p	RoutineStatusRecord#6+n+p	Hash_value#m [Byte#p]	C2	00-FF

[allg. Anf.: F-LAH_RxSWIN-881]

C1 = Nur vorhanden wenn ein Einzel-Hashwert verfügbar ist.

[allg. Anf.: F-LAH_RxSWIN-885]

C2 = Nur vorhanden wenn der Parameter [Type_of_hash] den Wert 0xFF hat und mehr als ein Hashwert verfügbar ist.

5.2.3.4 Positive Response Message Parameter Definition

[I: F-LAH_RxSWIN-883]

Folgende Parameter sind zu implementieren:

[allg. Anf.: F-LAH_RxSWIN-884]

Tabelle 5-22 Positive Response Message Parameter Definition

Definition
State_of_hashvalue 0x00 = Calculation_hashvalue_valid Die Berechnung des Hashw ertes w urde erfolgreich durchgeführt und der Hashw ert ist gültig. 0x01 = Calculation_hashvalue_invalid Die Berechnung des Hashw ertes ist fehlgeschlagen und der Hashw ert ist nicht gültig. 0x02 = Calculation_hashvalue_not_found Der Identifier für den Hashw ert ist nicht gültig und im DataIdentifier 0x0250 im Diagnose-Server nicht vorhanden. 0x03 - 0xFF = VOLKSWAGEN AG-reserved
Type_of_hashvalue 0x00 = VOLKSWAGEN AG-Reserved 0x01 = SHA-256, Die Berechnung des Hashw ertes erfolgt nach dem SHA-256-Verfahren. 0x02 - 0xFF = VOLKSWAGEN AG-reserved
Type_of_hash 0x00 = Einzel-Hashw ert Datensatz (bei Datensatzdow nload Gen. 1) 0x01 = Einzel-Hashw ert Bootloader-Datensatz (bei Datensatzdow nload Gen. 2) 0x02 = Einzel-Hashw ert Applikations-Datensatz (entspricht der Applikations-Datensatz-Nummer Datensatzdow nload Gen. 2) 0x03 = Einzel-Hashw ert über alle Anpassungen/Codierungen gemäß DataIdentifier 0x0250 0x04 - 0xFE = VOLKSWAGEN AG-reserved 0xFF = alle Einzel-Hashw erte gemäß DataIdentifier 0x0250
Individual_hash_value_id Dieser Parameter identifiziert Einzel-Hashw erte von Applikationsdatensätzen gemäß DataIdentifier 0x0250. 0x7200 - 0x72FF = w ird nur verw endet bei [Type_of_hash] = 0x02 0x0000 = bei allen anderen Werte im Parameter [Type_of_hashvalue] Alle anderen Werte sind VOLKSWAGEN AG-reserved

6 RxSWIN-spezifische Dokumentation

[I: F-LAH_RxSWIN-55]

Zu einer RxSWIN können mehrere "Steuergeräte-Versionen" mit ihren Identifikationsdaten und Integrity Validation Data zugeordnet sein. Die nachfolgenden Daten müssen mindestens zu jeder RxSWIN in den IT-Systemen dokumentiert werden.

6.1 Daten für die RxSWIN-spezifische Dokumentation eines DK3/DK3V/DK4/DK4V-System

[Prozess-Anf.: F-LAH_RxSWIN-149]

- Nur Gateway-Steuergeräte: Fahrgestellnummer (Vehicle Identification Number) aus DataIdentifier 0xF190
- Nur Gateway-Steuergeräte: Liste der RxSWINs aus dem DataIdentifier 0xF18F
- VW/Audi-Teilnummer (VW Spare Part Number) aus DataIdentifier 0xF187
- Softwareversion (VW Application Software Version Number) aus DataIdentifier 0xF189
- Hardwareteilnummer (VW ECU Hardware Number) aus DataIdentifier 0xF191
- optional: Hardwareversion (VW ECU Hardware Version Number) aus DataIdentifier 0xF1A3
- Teilnummer der/des Zieldatencontainer(s) (VW Data Set Number Or ECU Data Container Number)
- Version der/des Zieldatencontainer(s) (VW Data Set Version Number)
- FAZIT-Identifikation (FAZIT-Identification String) aus DataIdentifier 0xF17C
- Integrity Validation Data der Programmierung (Programming_hash) aus RoutineIdentifier 0x0253
- Integrity Validation Data der Konfiguration (Configuration_hash) aus RoutineIdentifier 0x0253
- Netzwerk_Adress_Information (N_AI)
- Knotenadresse (NA)
- Diagnoseadresse (DA)

6.2 Daten für die RxSWIN-spezifische Dokumentation eines SWCL

[Prozess-Anf.: F-LAH_RxSWIN-428]

- VW/Audi-Teilnummer (VW Spare Part Number) aus DataIdentifier 0xF187
- Softwareversion (VW Application Software Version Number) aus DataIdentifier 0xF189
- Teilnummer der/des Zieldatencontainer(s) (VW Data Set Number Or ECU Data Container Number)
- Version der/des Zieldatencontainer(s) (VW Data Set Version Number)
- Integrity Validation Data der Programmierung (Programming_hash) aus RoutineIdentifier 0x0253
- Integrity Validation Data der Konfiguration (Configuration_hash) aus RoutineIdentifier 0x0253
- Netzwerk_Adress_Information (N_AI)
- Diagnoseadresse (DA)
- Function-ID (F-ID)

6.3 Daten für die RxSWIN-spezifische Dokumentation eines DK2/DK2F/DK2FV-System

[Prozess-Anf.: F-LAH_RxSWIN-89]

- VW/Audi-Teilnummer (VW Spare Part Number) aus DataIdentifier 0xF187 bzw. 0x6200..63FF
- Hardwareteilnummer (VW ECU Hardware Number) aus DataIdentifier 0xF191 bzw. 6600..67FF
- Softwareversion (VW Application Software Version Number) aus DataIdentifier 0xF189 bzw. 0x6400..65FF
- Teilnummer der/des Zieldatencontainer(s) (VW Data Set Number Or ECU Data Container Number)
- Version der/des Zieldatencontainer(s) (VW Data Set Version Number)
- optional: Hardwareversion (VW ECU Hardware Version Number) aus DataIdentifier 0xF1A3 bzw. 6800..69FF
- optional: FAZIT-Identifikation (FAZIT-Identification String) aus DataIdentifier 0xF17C bzw. 6E00..6FFF
- nur bei DK2F/DK2FV: Integrity Validation Data der Programmierung (Programming_hash) aus DataIdentifier 0x0249 bzw. 0xA800..A9FF
- Integrity Validation Data der Konfiguration (Configuration_hash) aus DataIdentifier 0x0245 bzw. AA00..ABFF
- Netzwerk_Adress_Information (N_AI) des übergeordneten DK4-Systems
- SubSystemNodeAddress (SSN)
- Diagnoseadresse (DA)

7 Mitgeltende Dokumente und Spezifikationen

[I: F-LAH_RxSWIN-5]

/1/ LAH.DUM.909.G - Q-LAH 80124 - Unified Diagnostic Services Protocol (UDS), Emissions-related Diagnostic Services (legislated OBD)

[I: F-LAH_RxSWIN-48]

/2/ LAH.DUM.909.H - Q-LAH 80125 - Identifikation elektronischer Fahrzeugsysteme

[I: F-LAH_RxSWIN-49]

/3/ LAH.DUM.909.B - Q-LAH 80127 - Diagnose verteilter Systeme; Diagnoseanforderungen an Busmaster und Systeme

[I: F-LAH_RxSWIN-69]

/4/ LAH.DUM.907.BD - Q-LAH Schutz der Fahrzeugdiagnose (SFD)

[I: F-LAH_RxSWIN-75]

/5/ LAH.DUM.000.AD - Flashdatensicherheit für UDS Steuergeräte

[I: F-LAH_RxSWIN-76]

/6/ LAH.DUM.906.A - Q-LAH 80126 - UDS konforme Programmierung von Steuergeräten

[I: F-LAH_RxSWIN-144]

/7/ LAH.DUM.907.R1 - Datensatzdownload Generation 2

[I: F-LAH_RxSWIN-242]

/8/ LAH.DUM.907.BE - QLAH_Datenarten

[I: F-LAH_RxSWIN-540]

/9/ LAH Diagnosefilter

[I: F-LAH_RxSWIN-543]

/10/ Entfällt - Keine Beschreibung

[I: F-LAH_RxSWIN-596]

/11/ Datensatzdownload Generation 1

[I: F-LAH_RxSWIN-648]

/12/ LAH.000.036.G - Q-LAH 80127ES - Ergänzungsspezifikation für hochintegrierte Systeme

[I: F-LAH_RxSWIN-649]

/13/ LAH.000.036.H - Update filebasierter Systeme - Ergänzungsspezifikation für Non-Embedded-Systeme

[I: F-LAH_RxSWIN-650]

/14/ LAH.DUM.906.B - Q-LAH 80128 Teil 3 - Spezifikation für Flashcontainer ODX-Flash PDX-Flash

[I: F-LAH_RxSWIN-697]

/15/ LAH.000.900.AT - VKMS-Kernfunktionalität

[I: F-LAH_RxSWIN-702]

/16/ LAH.DUM.907.Q1 - Diagnoseschnittstelle SWaP

[I: F-LAH_RxSWIN-703]

/17/ FoD-LAH - Function-On-Demand Vehicle Requirements

[I: F-LAH_RxSWIN-704]

/18/ LAH.DUM.907.xx - Lastenheft Wegfahrsperre Master/Slave

[I: F-LAH_RxSWIN-705]

/19/ LAH.DUM.907.xx - Lastenheft Komponentenschutz Master/Slave

[I: F-LAH_RxSWIN-706]

/20/ LAH.DUM.900.AB - Hardware Security Module