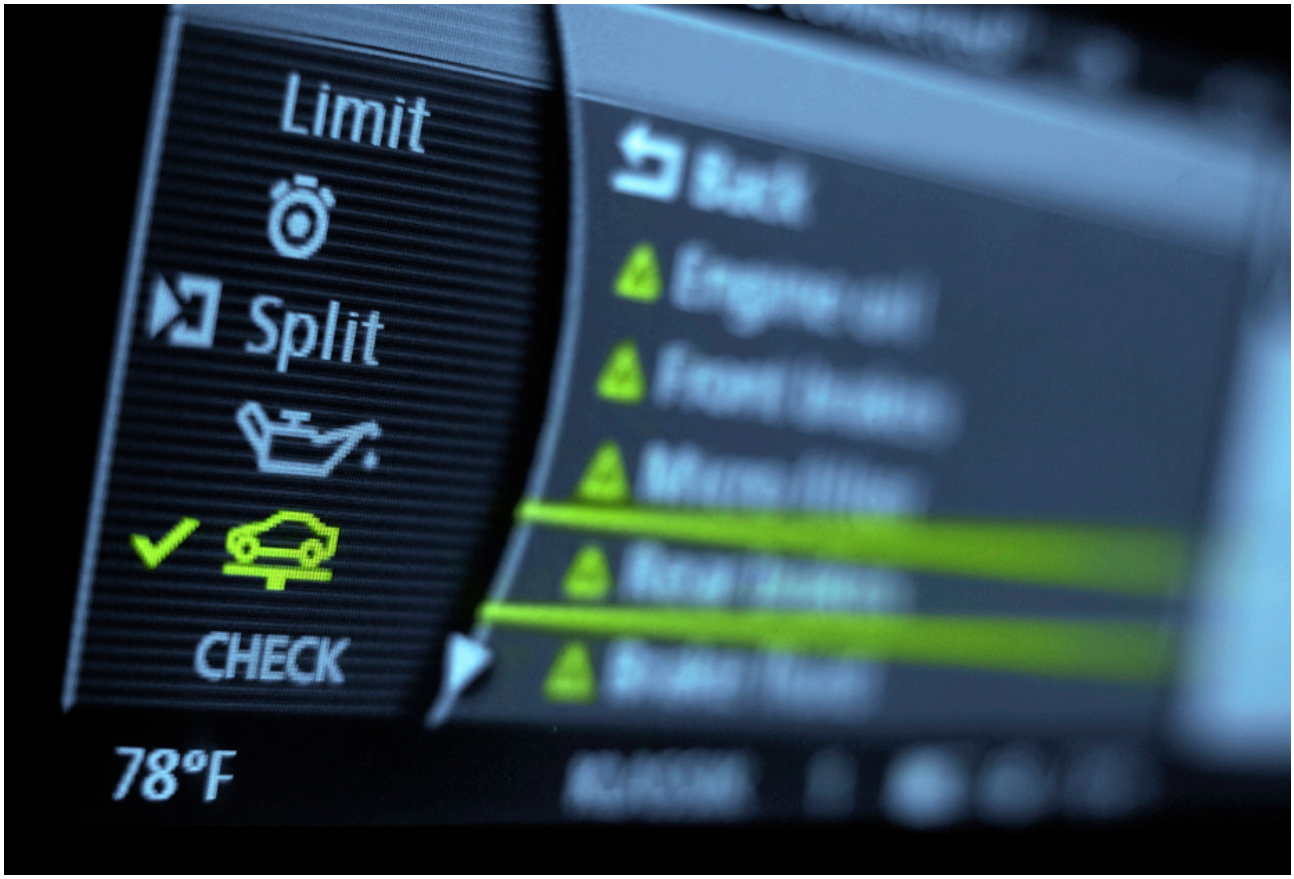
Elektrobit

EB tresos® AutoCore Generic 8

IP Stack documentation

release notes update for the SoAd module

product release 8.8.7



Elektrobit Automotive GmbH
Am Wolfsmantel 46
91058 Erlangen, Germany
Phone: +49 9131 7701 0
Fax: +49 9131 7701 6333
Email: info.automotive@elektrobit.com

Technical support

https://www.elektrobit.com/support

Legal disclaimer

Confidential information.

Table of Contents

1. Overview

This document provides you with the release notes to accompany an update to the `SoAd` module. Refer to the changelog Section 2.1, “Change log” for details of changes made for this update.

Release notes details

- EB tresos AutoCore release version: 8.8.7
- EB tresos Studio release version: 29.2.0
- AUTOSAR R4.2 Rev 2
- Build number: B577598

2. SoAd module release notes

- AUTOSAR R4.2 Rev 2
- AUTOSAR SWS document version: 4.2.2
- Module version: 1.8.25.B577598
- Supplier: Elektrobit Automotive GmbH

2.1. Change log

This chapter lists the changes between different versions.

Module version 1.8.25

2022-11-04

- Internal module improvement. This module version update does not affect module functionality.

Module version 1.8.24

2022-10-29

- Added new state SOAD_SOCON_WAITOFFLINE.

Module version 1.8.23

2022-10-28

- Removed direct dependency to Tls module.
- ASCSOAD-1679 Fixed known issue: SoAd resets remote address when connection is lost.

Module version 1.8.22

2022-09-16

- Improved TCP-TLS client connection establishment for SD SomeIP communication.

Module version 1.8.21

2022-06-10

- Implemented support for reporting security events to IdsM.
- Implemented handling FIN received with Tls.

Module version 1.8.20

2022-05-06

- Improved TLS API handling.

Module version 1.8.19

2022-02-18

- Added support for DTLS.

Module version 1.8.18

2021-10-08

- ASCSOAD-1534 Fixed known issue: SoConState is not changed to SOAD_SOCON_RECONNECT if Tx-Confirmation is handled for UDP SocketConnections.
- ASCSOAD-1545 Fixed known issue: SoAd disables wrong SoAdSocketRouteDestination.

Module version 1.8.17

2021-08-30

- ASCSOAD-1526 Fixed known issue: TP reception in non header mode may lead to endless loop.
- ASCSOAD-1538 Fixed known issue: SoAd enables wrong SoAdSocketRouteDestination.
- Added support for NPDU Buffer Pooling.

Module version 1.8.16

2021-06-25

- Added Support for Multiple Provided Service Instances.
- Improved error handling to keep TCP connection alive if transmission was aborted and no data was sent.
- ASCSOAD-1479 Fixed known issue: Out-of-bounds access may occur if TcpIp_TcpTransmit() returns a value other than TCPIP_OK.

Module version 1.8.15

2021-03-05

- ASCSOAD-1441 Fixed known issue: Incorrect Tx buffering of UDP frames.

Module version 1.8.14

2020-10-23

- ASCSOAD-1411 Fixed known issue: Closing of a UDP socket connection always closes the underlying TcpIp socket.

Module version 1.8.13

2020-06-19

- Improved performance of SoAd_IfTransmit().
- ASCSOAD-1337 Fixed known issue: SoAd generation error occurs due to PduLength bigger than 65535.
- ASCSOAD-1358 Fixed known issue: A socket connection group with TRIGGER_NEVER and TRIGGER_ALWAYS Tx PDUs may cause an out-of-bounds access.
- Added API SoAd_IsConnectionReady.
- ASCSOAD-1338 Fixed known issue: SoAdRouteMax set to INDEX_UINT8 can lead to incorrect mapping between PduRouteDestVirtualIds and SoCons.

Module version 1.8.12

2020-02-21

- ASCSOAD-1296 Fixed known issue: SoAd may not choose an ephemeral local port if SoAdSocketLocalPort is configured with zero.
- ASCSOAD-1312 Fixed known issue: Compilation error occurs when PostBuild variants are configured.

- ASCSOAD-1318 Fixed known issue: SoAdPduRoutes are not enabled correctly at initialization on big-endian platform.
- ASCSOAD-1323 Fixed known issue: TCP client may not immediately connect to TCP server.
- ASCSOAD-1306 Fixed known issue: SoAdRelocatableCfgEnable set to true can lead to wrong socket connection mapping.
- ASCSOAD-1313 Fixed known issue: Discarding of received frames can cause an out-of-bounds access.

Module version 1.8.11

2019-10-11

- ASCSOAD-1264 Fixed known issue: Java exception occurs if "SoAdSocketnPduUdpTxBufferMin" is enabled.
- ASCSOAD-1274 Fixed known issue: Wrong initialization of SoAd PBRAM.
- Added caching for XPath function getSoAdRoutesForPdu().
- Fixed IPv6 address detection logic in code generator.
- Improved file structure.
- ASCSOAD-1276 Fixed known issue: The reception of a UDP frame may cause an out-of-bounds access.
- ASCSOAD-1293 Fixed known issue: Reception ring buffer can cause an out-of-bounds access.

Module version 1.8.10

2019-09-06

- Improved performance of checks that verify consistency of SoAd and Sd configuration data.
- Improved performance of Routing Group Handling in SoAd for SoAd_IfTransmit.
- Added functionality to skip If Tx confirmation handling at PDU level.

Module version 1.8.9

2019-07-05

- ASCSOAD-1198 Fixed known issue: SoAd calls TcpIp APIs with invalid socket ID if socket gets closed while data are buffered.
- ASCSOAD-1215 Fixed known issue: Reception of TCP frame with length in header bigger than 64 KB may cause an out-of-bounds read access.

- ASCSOAD-1130 Fixed known issue: SoAd blocks transmission on TCP socket connection if TcpIp_TcpTransmit() returns a value other than E_OK.

Module version 1.8.8

2019-06-14

- ASCSOAD-1131 Fixed known issue: Remote address of first socket connection is locked with SoAd_SetUniqueRemoteAddr() instead of the matching socket connection in this socket connection group.
- ASCSOAD-1142 Fixed known issue: Unintended reset of remote address for Tcp and Udp socket connections if UdpSupervisionAliveTimeout is used for any socket connection.
- ASCSOAD-1173 Fixed known issue: Buffer overflow can be triggered on reception of an invalid frame.
- Improved SoAd_MainFunction() runtime at idle time.
- ASCSOAD-1199 Fixed known issue: SoAd_ReleaseRemoteAddr doesn't reset the remote address immediately if called by upper layer in context of SoAd_RxIndication().

Module version 1.8.7

2019-03-07

- ASCSOAD-1053 Fixed known issue: SoAd triggers SEGMENT FAULT on reception of TP PDUs on TCP connection with enabled header mode.
- Added full support of SoAd_ReleaseRemoteAddr(). Important note: This update of SoAd requires an ACG 8 Sd version 1.4.1 or later
- ASCSOAD-1105 Fixed known issue: TcpIpEvent() called with TCPIP_UDP_CLOSED sets transition change only for first SoCon in UDP SoConGroup.

Module version 1.8.6

2019-02-22

- ASCSOAD-1050 Fixed known issue: SoAd triggers SEGMENT FAULT on TP routing.

Module version 1.8.5

2019-02-15

- Added support of double buffering for concurrent If Tx PDUs using the same UDP socket.
- Added support to disable SoAdSocketTpRxBufferMin even for TCP connections and PDUs using Tp API.
- ASCSOAD-1050 Fixed known issue: No subscriptions are sent for SD OFFER messages received while client service is in DOWN state and socket connection is closed.
- Improved RAM usage by splitting internal data structure.

Module version 1.8.4

2019-01-24

- ASCSOAD-1057 Fixed known issue: SoAdSocketRoutes are not sorted consecutively based on header ID.

Module version 1.8.3

2018-12-13

- Added support for TLS extension.
- Added support measurement data.

Module version 1.8.2

2018-10-26

- ASCSOAD-901 Fixed known issue: SoAdTxUdpTriggerMode TRIGGER_NEVER does not work in combination with IP fragmentation.
- Added support for SoAd_IfTransmit() with SduDataPtr = NULL_PTR on a UDP connection to retrieve data with Up_[SoAd][If]TriggerTransmit().
- ASCSOAD-930 Fixed known issue: SoAd may corrupt the last TCP segment of a received PDU if header mode and IF API are used.

Module version 1.8.1

2018-09-20

- Added Post-build selectable support.
- ASCSOAD-922 Fixed known issue: Wrong PDU header ID may be inserted before the PDU is transmitted via a socket connection.

- Added Post-build binary support.

Module version 1.8.0

2018-06-22

- ASCSOAD-834 Fixed known issue: The nUdpPduBuffer is never triggered by timeout if SoAdTxUdpTriggerTimeout is less than SoAdMainFunctionPeriod.
- Separated the transmit part of SoAd_MainFunction to SoAd_MainFunctionTx and added an option to make the latter externally callable.
- Optimized handling of client/server configuration using meta data handling.

Module version 1.7.24

2018-05-30

- ASCSOAD-823 Fixed known issue: Updating LAST-IS-BEST PDU could lead to buffer inconsistency.
- Internal module improvement. This module version update does not affect module functionality.

Module version 1.7.23

2018-03-16

- ASCSOAD-813 Fixed known issue: A UDP Tx buffer is provided only for the first socket connection if a PDU route destination refers to the socket connection group.

Module version 1.7.22

2018-02-16

- Implemented N:1 routing.
- ASCSOAD-778 Fixed known issue: SoAd sends incorrect data if SoAdTxPduCollectionSemantics is set to last-is-best.
- ASCSOAD-787 Fixed known issue: Incorrect configuration check for number of configured UDP sockets for IPv6.
- ASCSOAD-788 Fixed known issue: Incorrect configuration check for number of configured TCP sockets for IPv6.

Module version 1.7.21

2017-12-15

- Internal module improvement. This module version update does not affect module functionality.

Module version 1.7.20

2017-11-17

- Updated TpTxConfirmation() to trigger another transmission for the same PDU.

Module version 1.7.19

2017-09-22

- Added support for SoAdTxPduCollectionSemantics(last-is-best vs. queued).
- Added API SoAd_ReleaseRemoteAddr() to reset the remote address.
- Updated to MISRA 2012
- ASCSOAD-640 Fixed known issue: Message lost or corruption with multiple PduRouteDest + nPduUdp-TxBuffer + TRIGGER_NEVER.

Module version 1.7.18

2017-08-25

- Internal module improvement. This module version update does not affect module functionality.

Module version 1.7.17

2017-07-28

- Internal module improvement. This module version update does not affect module functionality.

Module version 1.7.16

2017-06-30

- Time consumption in matching InternalRoutingGroup and ExternalRoutingGroup.

- ASCSOAD-638 Fixed known issue: SoAd might reject API calls for specific routing groups.

Module version 1.7.15

2017-05-05

- Added support of post build RAM greater than 64kB.

Module version 1.7.14

2017-03-03

- ASCSOAD-605 Fixed known issue: Invalid memory access occurs if a single routing group is configured.

Module version 1.7.13

2017-02-03

- Internal module improvement. This module version update does not affect module functionality.

Module version 1.7.12

2016-12-14

- ASCSOAD-572 Fixed known issue: A global routing group is treated as specific routing group if reference to SocketConnectionGroup exists for this routing group.
- ASCSOAD-588 Fixed known issue: SoAd sends frames to wrong destination if initiated by SoAd_IfSpecificRoutingGroupTransmit().

Module version 1.7.11

2016-12-02

- Internal module improvement. This module version update does not affect module functionality.

Module version 1.7.10

2016-11-04

- Updated handling of TcpIp_ReturnType according to AUTOSAR 4.2.2.

Module version 1.7.9

2016-10-07

- Replaced config parameter SoAdSoConMax and SoAdRoutingGroupMax with SoAdSoConIdType and SoAdRoutingGroupIdType.
- ASCSOAD-527 Fixed known issue: SoAd tries to request already provided data from Tp upper layer if UDP retry is performed.
- SoAd_CloseSoCon() called with Abort = FALSE shall not stop ongoing Rx/Tx.
- ASCSOAD-526 Fixed known issue: Trigger transmit requests are handled incorrect if specific routing groups are triggerable and have multiple socket connections.
- Added support for TcpIp config schema according to AUTOSAR 4.2.2.

Module version 1.7.8

2016-09-09

- ASCSOAD-499 Fixed known issue: No UDP retry might performed if buffer gets overwritten by newer frame.
- Added Udp alive supervision feature.
- ASCSOAD-500 Fixed known issue: Reception of invalid header values leads to invalid memory access.
- ASCSOAD-501 Fixed known issue: SoAd fills segmented Rx TCP PDUs with wrong data.
- ASCSOAD-502 Fixed known issue: SoAd transmits invalid payload.

Module version 1.7.7

2016-08-05

Module version 1.7.6

2016-07-01

- ASCSOAD-449 Fixed known issue: The call of SoAd_CloseSoCon() closes an active TCP connection with RST instead of FIN.

- ASCSOAD-454 Fixed known issue: Preemption of SoAd_IfTransmit() causes a copy attempt from NULL_PTR.

Module version 1.7.5

2016-05-25

- ASCSOAD-433 Fixed known issue: TcpIp_Close() is called twice for the listen socket.
- ASCSOAD-425 Fixed known issue: The call of SoAd_CloseSoCon() and SoAd_TcpIpEvent(TCPIP_TCP/UDP_CLOSED) within the same main function triggers TcpIp_Close() with invalid socket ID.

Module version 1.7.4

2016-04-29

- Updated memory section macros to AUTOSAR 4.0 naming convention.
- ASCSOAD-414 Fixed known issue: SoAdPduRoute of specific routing groups try to send over the first socket connection in group only.
- ASCSOAD-416 Fixed known issue: SoAd might confirm successful TP interface transmissions with error code NTFRSLT_E_NOT_OK.
- ASCSOAD-418 Fixed known issue: SoAd writes to wrong memory location when Rx only socket connection is closed.

Module version 1.7.3

2016-04-01

- Added support for Debug & Trace with custom header file configurable via parameter `BaseDbgHeaderFile`.
- ASCSOAD-388 Fixed known issue: SoAd does not handle a return value other than E_OK for TcpIp_Bind(), TcpIp_TcpListen(), and TcpIp_TcpConnect() correctly.
- ASCSOAD-399 Fixed known issue: SoAd rejects API calls for specific routing groups.

Module version 1.7.2

2016-02-05

- Added check when a SoAdRoutingGroup is not referenced by any SoAdSocketRoute or SoAdPduRoute.

- Implemented support for connection specific TCP Keep-Alive configuration.

Module version 1.7.1

2015-11-06

- ASCSOAD-323 Fixed known issue: SoAd might not be able to reestablish a TCP connection if it was previously closed by the remote node.
- ASCSOAD-325 Fixed known issue: Inconsistent SoAd configuration might lead to EB tresos Studio errors.
- ASCSOAD-326 Fixed known issue: SoAd might enable, disable or trigger a different SoAdRoutingGroup then intended.
- ASCSOAD-322 Fixed known issue: SoAd calls of TcpIp_DhcpRead/WriteOption() pass wrong parameter formats and wrong option values.
- ASCSOAD-334 Fixed known issue: SoAd_ChangeParameter() calls TcpIp_ChangeParameter() with wrong SocketId parameter value.
- ASCSOAD-333 Fixed known issue: SoAd might close wrong socket instead of listen socket.
- ASCSOAD-335 Fixed known issue: The SoAd incorrectly reports a configuration error if all SoAdRoutingGroups are triggerable.
- Created recommended configuration.
- ASCSOAD-340 Fixed known issue: The call of SoAd_LocalIpAddrAssignmentChg() with TcpIpAddrId which is not referred by SoAd leads to DET error report if enabled or might cause invalid memory access if disabled.
- ASCSOAD-342 Fixed known issue: SoAd uses wrong destination port for outgoing UDP/IPv6 datagrams.

Module version 1.7.0

2015-06-19

- Changed internal type for TcpIp_SocketIdType from uint8 to uint16, thus supporting TcpIp-stacks providing socketIds larger than 255.
- Added Seamless Service Relocation Support.

Module version 1.6.5

2015-02-20

- ASCSOAD-296 Fixed known issue: SoAd might stop forwarding received data on TCP connections.

- ASCSOAD-297 Fixed known issue: SoAd might not recover from an aborted TCP connection attempt.

Module version 1.6.4

2014-10-02

- Added support of IPv6.
- Changed module to include only TcpIp.h as TcpIp module interface file.
- ASCSOAD-284 Fixed known issue: SoAd might perform an invalid memory access in case upper layers use indirect data provision.

Module version 1.6.3

2014-04-25

- Added support of large (length > 255 Byte) PDU transmission via `IfRoutingGroupTransmit`.

Module version 1.6.2

2013-10-11

- ASCSOAD-249 Fixed known issue: SoAd might not acknowledge TCP data during disconnection and delay the closure procedure.
- ASCSOAD-254 Fixed known issue: SoAd might transmit invalid `PduHeader` information on TCP socket connections that use the Nagle algorithm.
- Added support of `nPduUdpTxBuffer` functionality.

Module version 1.6.1

2013-06-14

- Internal module improvement. This module version update does not affect module functionality.

Module version 1.6.0

2013-04-08

- ASCSOAD-220 Fixed known issue: SoAd might perform an invalid reading memory access if a socket connection is only configured for transmissions.

- ASCSOAD-221 Fixed known issue: `SoAd_SetRemoteAddr()` refuses the request for open UDP socket connections with disabled PDU header mode.
- Added support for multiple PDU routing group references in `SocketRoutes` and `PduRoutes`.
- ASCSOAD-232 Fixed known issue: SoAd configuration does not allow UDP and TCP `SocketConnection` with the identical local address and local port.
- Added support of PDU transmission via `IfRoutingGroupTransmit` API.
- ASCSOAD-237 Fixed known issue: SoAd might not receive UDP datagrams in `SoAdSocketConnectionGroups` with more than one `SoAdSocketConnection`.

Module version 1.5.0

2013-02-20

- ASCSOAD-175 Fixed known issue: SoAd might not compile if an upper layer provides only one of the notification APIs `SoAdSoConModeChg` and `SoAdLocalIpAddrAssigmentChg`.
- ASCSOAD-176 Fixed known issue: SoAd might not reopen a TCP listen socket connection correctly.
- ASCSOAD-186 Fixed known issue: `SoAd_GetLocalAddr` does not provide local port information.
- Added support of multiple UDP `SocketConnections` per `SocketConnectionGroup`.
- ASCSOAD-209 Fixed known issue: SoAd might not correctly process multiple PDUs within one received frame.
- Added support of PDU routing groups.
- Updated AUTOSAR SWS SocketAdaptor 2.0.24 R4.1 Rev 1.
- Implemented provision of `<Up>_[SoAd][Tp]TxConfirmation()` for UDP transmissions in the context of `SoAd_MainFunction`.
- ASCSOAD-211 Fixed known issue: SoAd might transmit invalid data if the UDP retry feature is enabled.
- Added support of TCP immediate TP transmit confirmation.

Module version 1.4.1

2012-10-16

- ASCSOAD-144 Fixed known issue: SoAd might perform an invalid memory access in case of a UDP socket connection for which no TP receive buffer is configured.
- ASCSOAD-147 Fixed known issue: Upper Layer of SoAd might not receive the full TP-PDU in case of a UDP socket connection.
- ASCSOAD-152 Fixed known issue: Upper Layer of SoAd might not receive the full TP-PDU in case the upper layer accepts only a part of the TP-PDU.

- ASCSOAD-153 Fixed known issue: Upper layer of SoAd might receive invalid data in case the upper layer accepts only a part of the TP-PDU and a new TP-PDU is received by SoAd in the meantime.
- ASCSOAD-155 Fixed known issue: SoAd might not be able to reopen a socket connection if it was closed by TcpIp module with `TCPIP_TCP_RESET`.
- Added qualifier `const` to parameter `PduInfoPtr` of `CopyRxData`.
- Updated AUTOSAR SWS SocketAdaptor 2.0.19 R4.1 Rev 0

Module version 1.4.0

2012-09-18

- Updated TcpIp types (`TcpIp_ParamIdType`, `TcpIp_DomainType`, `TcpIp_ProtocolType`).
- Added `TxQuota` functionality.
- Added name macros for configuration parameters which have `SYMBOLICNAMEVALUE` set to true.

Module version 1.3.1

2012-07-31

- Updated PDU fan-out TX (i.e. more than one `SoAdPduRouteDest` per `SoAdPduRoute`).

Module version 1.3.0

2012-07-13

- ASCSOAD-115 Fixed known issue: TCP socket connection with PDU header mode disabled might be blocked for upper layers with TP-API.
- Updated signature of `TcpIp_UdpTransmit` and `TcpIp_TcpTransmit`.
- Updated SoAd config to SWS 2.0.13.

Module version 1.2.2

2012-06-27

- Added support of immediate shutdown with `SoAd_CloseSoCon`.
- Implemented SoAd extension for better support of `UdpNm`.
- Added support of `UdpRetry` functionality.

- Added support of multiple TCP `SocketConnections` per `SocketConnectionGroup`.
- ASCSOAD-107 Fixed known issue: `SoAdTp_Transmit()` cannot be called within `<Up>_[SoAd][Tp]TxConfirmation()` for the same PDU related to a TCP socket connection.
- ASCSOAD-108 Fixed known issue: SoAd might handle socket connections incorrectly if the related socket is closed by TcpIp.

Module version 1.2.1

2012-06-01

- Added SoAd TP Rx-PDU cancellation support.
- ASCSOAD-86 Fixed known issue: SoAd requires transmit confirmation enabled for upper layers with If-API.

Module version 1.2.0

2012-05-21

- Added SoAd TP support.
- Updated SoAd config to SWS 2.0.9.

Module version 1.1.0

2012-04-13

- Internal module improvement. This module version update does not affect module functionality.

Module version 1.0.0

2012-03-16

- Initial AUTOSAR 4.0 version.

2.2. New features

- Handling FIN received with Tls.

  Implemented support for reporting security events to IdsM.

2.3. Elektrobit-specific enhancements

This chapter lists the enhancements provided by the module.

- Configurable reporting of `SOAD_E_INV_PDUHEADER_ID` to DET

  Description:

  A pre-compile time configuration parameter is provided that enables/disables the reporting of `SOAD_E_INV_PDUHEADER_ID` to DET.

- `SoAd_GetLocalAddr()` allows to retrieve the address family

  Description:

  The call of `SoAd_GetLocalAddr()` with domain of the local address set to TCPIP_AF_UNSPEC will return the domain of the configured address family.

- API support of `SoAd_ReleaseRemoteAddr()` to reset the remote address.

  Description:

  The call of `SoAd_ReleaseRemoteAddr()` can be used to reset the remote address of a UDP connection immediately to configured value.

- Support to increase Tx Tp frame processing

  Description:

  `SoAd` provides `SoAdEnableMainFunctionTx` which enables the TP transmit section of `SoAd_MainFunction()` to be callable externally via `SoAd_MainFunctionTx()`. Additional calls of `SoAd_MainFunctionTx()` speed up the Tx Tp frame processing.

- Support of double buffering Udp Tx If frame

  Description:

  `SoAd` uses the nPDU buffer `SoAdSocketnPduUdpTxBufferMin` to store incoming Udp If PDUs if it interrupts a transmission on the corresponding Udp socket. In case that the interrupt occurs during transmission of current nPDU buffer, the remaining buffer space is used as temporary buffer.

  Rational:

  According to https://bugzilla.autosar.org/show_bug.cgi?id=59416 `TcpIp_UdpTransmit()` is not reentrant for the same UDP socket. If Tx PDUs for the same Udp socket interrupt each other, SoAd needs either to reject the Tx request or to buffer them.

- Support of Udp Tx If PDU bigger than buffer

  Description:

If a Udp Tx If PDU does not fit into the buffer `SoAd` will try to transmit it. In case that transmission got rejected by `TcpIp`, then `SoAd` will reject the PDU.

- Support to disable `SoAdSocketTpRxBufferMin` even for TCP connections and PDUs using Tp API

  Description:

  If upper layer always accepts received data for a TCP connection and PDUs with Tp API, it is no longer mandatory to configure a reception buffer. The data will be provided in context of `SoAd_RxIndication()`. In case that no buffer is configured and upper layer rejects data reception,then either the PDU gets dropped if header mode is disabled or the TCP connection gets closed if header mode is disabled.

- Disabling of If Tx confirmation handling

  Description:

  The If Tx confirmation handling is a time consuming task which can affect the performance. This can be optimized either by disabling Tx confirmation for a complete upper layer by disabling the config parameter `SoAdBswModules/SoAdIfTxConfirmation` or at PDU level by disabling the config parameter `SoAdConfig/SoAdPduRoute/SoAdSkipIfTxConfirmation`. Both measures reduce the main function execution time.

- SoAd_IsConnectionReady API

  Description:

  The call of `SoAd_IsConnectionReady()` can be used to check if an ARP entry or IpSec SA exists for this socket connection.

2.4. Deviations

This chapter lists the deviations of the module from the AUTOSAR standard.

- Unsupported SoAd Features

  Description:

  SoAd does not yet support the following features:

  - Resource Management
  - Udp StrictHeaderLen Check
  - Socket connection notification enabling/disabling

  SWS_SoAd_00649, SWS_SoAd_00125, SWS_SoAd_00126, SWS_SoAd_00597

- Initialization check in main function

Description:

If the main function is called while the module is not yet initialized, the main function returns immediately without performing any functionality and without raising any Det error. This initialization check is always performed independent of the development error detection setting.

Rationale:

The SchM module may schedule the modules main function before the module is initialized. This would result in lots of Det errors during start up. Therefore the module's main function does not throw a Det error if the module is not yet initialized and simply returns in this case.

- Configuration check

Description:

`SoAd_Init()` does not check the `SoAdConfigPtr` for containing a valid configuration. Instead it will perform a basic `NULL_PTR` check.

Rationale:

SoAd configuration does not include signatures to verify if a configuration is valid or not A valid configuration is expected.

SWS_SoAd_00216

- Symbolic name for `SoAdSocketId` and `SoAdRxPduId` do not follow the AUTOSAR naming scheme

Description:

The symbolic name macros for `SoAdSocketId` and `SoAdRxPduId` do not follow the AUTOSAR 4.0 Rev 3 naming scheme. AUTOSAR only specifies about the inclusion of the name of direct parent in the symbolic macros but instead the macros are generated as follows: `SoAdSocketId`: `SoAdConf_[SoAdSocketConnectionGroup]_[SoAdSocketConnection]` `SoAdRxPduId`: `SoAdConf_[SoAdSocketRoute]_[SoAdSocketRouteDest]`

Rationale:

`SoAdSocketConnection` short names are only distinct within the context of the superior `SoAdSocketConnectionGroup`. The generation of `SymbolicName` macros as specified within the ECU configuration specification could lead to multiple macro redefinitions.

`SoAdSocketRouteDest` short names are only distinct within the context of the superior `SoAdSocketRoute`. The generation of `SymbolicName` macros as specified within the ECU configuration specification could lead to multiple macro redefinitions.

- SoAd does not support `<Up>_[SoAd][Tp]StartOfReception` API according to AUTOSAR 4.1.1 and higher

Description:

SoAd does not support the extended `StartOfReception` API that was introduced in all TP related modules in AUTOSAR version 4.1.1. The `StartOfReception` API was extended with the new parameter `PduInfoType*` info.

Rationale:

This deviation is required for the compatibility to AUTOSAR version prior to 4.1.1. Thus RFC https://www.autosar.org/bugzilla/show_bug.cgi?id=47264 is not considered.

- `SoAd_LocalIpAddrAssignmentChg()` silently ignores invalid a call with not expected `TcpIpAddrId`

Description:

If TcpIp calls `SoAd_LocalIpAddrAssignmentChg()` with not expected `TcpIpAddrId`, the function shall not call Det but silently ignore the call.

Rationale:

TcpIp calls `SoAd_LocalIpAddrAssignmentChg()` for all configured `TcpIpLocalAddr` even if SoAd does not refer to it. SoAd tolerate such calls and silently ignore not expected IDs. See also https://www.autosar.org/bugzilla/show_bug.cgi?id=71116.

Requirements:

SWS_SoAd_00280

- `RequestIpAddrAssignment()` syntax according to AUTOSAR 4.2.2 not supported

Description:

The syntax of `SoAd_RequestIpAddrAssignment()` does not contain the parameters `Netmask`, `DefaultRouterPtr` introduced by AUTOSAR 4.2.2. SoAd also calls `TcpIp_RequestIpAddrAssignment()` with syntax based on AUTOSAR 4.1.1. Thus RFC https://www.autosar.org/bugzilla/show_bug.cgi?id=62672 is not not considered.

Rationale:

The syntax of `RequestIpAddrAssignment()` based on AUTOSAR 4.1.1 is kept for backwards compatibility.

Requirements:

SWS_SoAd_00520

- `SoAd_TpChangeParameter()` API not supported

Description:

The API SoAd_TpChangeParameter() is not supported. Instead the EB specific API SoAd_ChangeParameter() can be used to change TcpIp specific parameters.

Requirements:

SWS_SoAd_00508, SWS_SoAd_00630, SWS_SoAd_00631

- A socket connection leaving the state SOAD_SOCON_OFFLINE always reports SOAD_SOCON_RECONNECT

Description:

If during the call of SoAd_MainFunction() a socket connection leaves the state SOAD_SOCON_OFFLINE, the module always calls Up_SoConModeChg() with the state SOAD_SOCON_RECONNECT to the upper layer. If the precondition are fulfilled to reach SOAD_SOCON_ONLINE, the module calls Up_SoConModeChg() again within the same SoAd_MainFunction().

Rationale:

The SoAd_MainFunction() can handle multiple socket connection state transition within one call. Therefore, direct transition from SOAD_SOCON_OFFLINE to SOAD_SOCON_ONLINE does not provide any benefit but would required extra treatment.

Requirements:

SWS_SoAd_00591

- SoAdSocketTcpNoDelay not supported.

Description:

The API TcpIp_ChangeParameter() will not be called to change TCPIP_PARAMID_TCP_NAGLE when allocating a new socket.

Requirements:

SWS_SoAd_00689

- SoAd supports AUTOSAR 4.0.3 EcuC references only.

Description:

SoAd only supports EcuC references according to AUTOSAR 4.0.3. The container EcucConfgSet is not part of the EcuC path.

Rationale:

EcuC is based on AUTOSAR 4.0.3 to stay compatible to other AUTOSAR 4.0.3 modules. Thus RFC https://www.autosar.org/bugzilla/show_bug.cgi?id=53369 is not considered.

Requirements:

ECUC_SoAd_00038, ECUC_SoAd_00030

- `SoAd` only support `<Up>_[SoAd][If]RxIndication` API according to AUTOSAR 4.1.2

Description:

`SoAd` uses the API syntax of AUTOSAR version 4.1.2 of `<Up>_[SoAd][If]RxIndication` which does not require a constant pointer for the parameter `PduInfoPtr`.

Rationale:

This deviation is required for the compatibility to upper layer modules with AUTOSAR version prior to 4.1.3.

Requirements:

SWS_SoAd_00106

- Configuration parameters `SoAdSoConMax` and `SoAdRoutingGroupMax` unused.

Description:

The configuration parameters `SoAdSoConMax` and `SoAdRoutingGroupMax` obsolete.

Rationale:

`SoAd` allows to change the number of socket connection and routing groups at post build time without the need of limits at precompile time. The number is only limited by the used type as well as the reserved post build RAM.

Requirements:

ECUC_SoAd_00127, ECUC_SoAd_00126, SWS_SoAd_00518, SWS_SoAd_00519

- Initialization check in main function

Description:

If the main function is called while the module is not yet initialized the main function returns immediately without performing any functionality and without raising any Det error. This initialization check is always performed independent of the development error detection setting.

Rationale:

The RTE module may schedule the modules main function before the module is initialized. This would result in lots of Det errors during start up. Therefore the module's main function does not throw a Det error if the module is not yet initialized and simply returns in this case.

Requirements:

SWS_SoAd_00283

- Limited Det error reporting

  Description:

  In the context of `SoAd_RxIndication()` the module only reports `SOAD_E_INV_ARG` if the parameter `RemoteAddrPtr` is NULL_PTR. In case that the remote address is valid but there is no match, the frame will be discarded.

  Requirements:

  SWS_SoAd_00268

- Pdu fan out not supported for TP

  Description:

  Fan out(i.e. multiple SoAdPduRouteDest specified) is only supported for upper layers with IF APIs.

  Requirements:

  SWS_SoAd_00561

- SoAd does not support IP fragmentation in combination with functionality Last-Is-Best

  Description:

  SoAd does not support Ip fragmentation in combination with Last-Is-Best nPDU transmission.

  Requirements:

  SWS_SoAd_00543

- `SoAd` supports `SoAdSocketUdpAliveSupervisionTimeout` handling according to AUTOSAR 4.3.0

  Description:

  In order to support "Release of unused socket connections" functionality, `SoAd` starts the Udp alive supervision timer according to AUTOSAR 4.3.0 requirements `SWS_SoAd_00694` and `SWS_SoAd_00742` instead of AUTOSAR 4.2.2 requirements `SWS_SoAd_00694`. Therefore, the Udp alive timer is started and updated after each received Udp frame which passes the message acceptance filter.

  Rationale:

  This deviation is required to support the "Release of unused socket connections" functionality. Otherwise, no Udp alive timer is started if a method call is received after the remote address was set with `SoAd_SetUniqueRemoteAddr()`. Hence, the call of `SoAd_ReleaseRemoteAddr()` would immediately release the remote address and interrupt an ongoing method call.

Requirements:

SWS_SoAd_00694

- `SoAd` does not forward the parameter `Abort` for UDP sockets.

Description:

`SoAd` does not forward the parameter `Abort` from `SoAd_CloseSoCon()` to `TcpIp_Close()` for UDP sockets. Instead `Abort` will always be FALSE for `TcpIp_Close()` of UDP sockets.

Rationale:

In case of UDP the call of `TcpIp_Close()` belongs to a group of socket connection. Only the last socket connection which gets closed will call `TcpIp_Close()`. Passing the parameter `Abort` of `SoAd_CloseSoCon()` call from the last socket connection would not reflect the other socket connections. Furthermore, TcpIp only specifies the usage of `Abort` for TCP sockets and ignores `Abort` for UDP sockets.

Requirements:

SWS_SoAd_00642

- State transition for UDP socket connection of type automatic with wildcard remote address is handled in next `SoAd_MainFunction` instead of directly before transmit confirmation function is called.

Description:

For a UDP socket connection of type automatic i.e. configuration parameter `SoAdSocketAutomaticSoConSetup` set to `TRUE`) which uses a wildcard in the configured remote address (i.e. an ANY-String for IP address or port), `SoAd` is not changing the state of the socket connection to `SOAD_SOCON_RECONNECT` directly before the related transmit confirmation function is called (or would be called if such a function is not configured). Instead the state transition is handled in the next `SoAd_MainFunction`.

Rationale:

To improve run-time performance and avoid preemption issues, the state transition will be handled in the next `SoAd_Mainfunction` after transmit confirmation function is called.

Requirements:

SWS_SoAd_00582

- SoAd_TcpIpEvent() does not report a Det for event unknown to SoAd.

Description:

If SoAd_TcpIpEvent() is called with an event that is currently not known to SoAd, the module no longer reports a Det and ignores the unknown event.

Rationale:

This allows lower layers to report more events than currently required by SoAd level and ensures easy future usage.

Requirements:

SWS_SoAd_00278

- TCP stream handling in header mode

Description:

Point 4 of requirement SWS_SoAd_00559 states "If the remainder is smaller than a PDU Header or the indicated length within the header SoAd shall stop processing and ignore the rest of the message." without any statement about the protocol. This sentence is only considered for UDP if SWS_SoAd_00709 does not apply.

Rationale:

Following this requirement part also for TCP would stay in contradiction with segmentation of the TCP stream including the point 1 of the same requirement. Issue is addressed with https://jira.autosar.org/browse/AR-116232.

Requirements:

SWS_SoAd_00559

- TCP stream handling in non mode

Description:

For upper layer using TP API (like DoIP) Up_[SoAd][Tp]StartOfReception() is called with TpSduLength = 0U to indicate a PDU reception with unknown length as defined by SWS_SoAd_00595. Each TCP segment is forwarded via Up_[SoAd][Tp]CopyRxData(). Up_[SoAd][Tp]RxIndication() is called when the TCP connection is closed. Furthermore, data which are rejected from upper layer or are too big to be stored in the reception buffer can not be discarded because it will break the stream. Sender and receiver will become out of sync. Therefore, SoAd closes the connection. SWS_SoAd_00563 is only valid for UDP connections with upper layer using IF and TP API and TCP connections with upper layer using IF API.

Rationale:

A TCP connection provides a reliable stream with receive window as flow control measure. If sender produces data which has to be discarded on the receiver side then it need to be handled by an higher protocol which parses the data. If the receiver rejects a part of the received stream it loses the synchronization point for parsing the stream which can only be solved via reconnection indicating the sender that the synchronization got lost.

Requirements:

SWS_SoAd_00563, SWS_SoAd_00568

- Support of PDUs with unknown header ID and for inactive RoutingGroups on a TCP connection

  Description:

  PDUs with unknown header ID and for inactive RoutingGroups on a TCP connection are only supported if they are transmitted unsegmented.

  Rationale:

  See chapter Limitations.

  Requirements:

  SWS_SoAd_00559, SWS_SoAd_00600

- SoAd_CloseSoCon() triggers that a socket connection immediately leaves the state ONLINE.

  Description:

  If SoAd_CloseSoCon() is called for the last user of a socket connection in state ONLINE, the socket state immediately changes to SOAD_SOCON_WAITOFFLINE in context of this function call. This intermediate stated is not propagated via Up_SoConModeChg() but will be returned when SoAd_GetSoConMode() is called.

  Rationale:

  Although SWS_SoAd_00588 defines to just remember the request to close in context of SoAd_CloseSoCon() and perform the closure in context of SoAd_MainFunction(), it is problematic for a socket connection user to identify for a close/open sequence (as described in chapter "7.1.3 Socket Connection Open/Close Sequence Remarks") if a connection is still in state ONLINE or again ONLINE when polling the state with SoAd_GetSoConMode(). SOAD_SOCON_WAITOFFLINE closes this gap.

  Requirements:

  SWS_SoAd_00512, SWS_SoAd_00588, SWS_SoAd_00597, SWS_SoAd_00741, SWS_SoAd_00767

2.5. Limitations

This chapter lists the limitations of the module. Refer to the module references chapter *Integration notes*, subsection *Integration requirements* for requirements on integrating this module.

- No support for multiple configurations

Description:

Only one configuration is supported. Multiple configurations are not allowed.

- Reception of a segmented PDU via a disabled `SoAdSocketRoute` causes reconnection

  Description:

  The reception of a segmented PDU belonging to a disabled `SoAdSocketRoute` (i.e. all related `SoAdRoutingGroups` are disabled) triggers the TCP socket connection to close and reopen within the next `SoAd_MainFunction()`. The closure is done by sending a `FIN` without acknowledging the rejected data and under consideration of `TcpIpTcpFinWait2Timeout`. It is not expected that PDUs for a disabled `SoAdSocketRoute` are exchanged over a point to point reliable TCP connection. Discarding of segmented PDUs is not supported.

- Reception of a segmented PDU with unknown header ID causes reconnection

  Description:

  The reception of a segment PDU with unknown header ID triggers the TCP socket connection to close and reopen within the next `SoAd_MainFunction()`. The closure is done by sending a `FIN` without acknowledging the rejected data and under consideration of `TcpIpTcpFinWait2Timeout`. It is not expected PDUs with unknown header IDs are exchanged over a point to point reliable TCP connection. Discarding of segmented PDUs is not supported.

- Reception of a segmented PDU exceeding the configured PduLengthTypeEnum + 8 byte header causes reconnection

  Description:

  The reception of a segment PDU with total length in header exceeding the configured PduLengthTypeEnum + 8 byte header triggers the TCP socket connection to close and reopen within the next `SoAd_MainFunction()`. The closure is done by sending a `FIN` without acknowledging the rejected data and under consideration of `TcpIpTcpFinWait2Timeout`. It is not expected that PDUs which are greater than the representation type are exchanged over a point to point reliable TCP connection. Discarding of segmented PDUs is not supported.

- `SoAdSocketTcpKeepAliveProbesMax` is limited to uint8

  Description:

  `SoAdSocketTcpKeepAliveProbesMax` can have values from 1 to 255 only.

- Only one TcpIp client is allowed for the same local IP address and port.

  Description:

  If `SoAdSocketTcpInitiate` is set to true, the socket connection group can only contain one socket connection.

- Linking of socket routes to TCP socket connection group is not supported.

  Description:

  It is not supported to link `SoAdRxSocketConnOrSocketConnBundleRef` to a `SoAdSocketConnectionGroup` with `SoAdSocketProtocol = SoAdSocketTcp` Only direct linkage to SoAdSocketConnection are supported.

- SoAd_IfTransmit() with SduDataPtr = NULL_PTR is not supported for TCP connections.

  Description:

  SoAd does not support calls of SoAd_IfTransmit() with SduDataPtr = NULL_PTR for TCP connections. In this case a Det is reported.

- `SoAdGetAndResetMeasurementDataApi` is not supported in ACG 8.5.2

  Description:

  The functionality of `SoAdGetAndResetMeasurementDataApi` is not supported in ACG 8.5.2. The config parameter shall not be enabled.

- `SoAdTlsEnabled` is not supported in ACG 8.5.2

  Description:

  The functionality of `SoAdTlsEnabled` is not supported in ACG 8.5.2. The config parameter shall not be enabled.

- Pdu length maximum if PduHeaderMode is enabled.

  Description:

  To support buffering of PDUs if PduHeaderMode is enabled, the Pdu length is limited to PDULengthTypeMAX - PduHeaderLength.

2.6. Open-source software

SoAd does not use open-source software.