Elektrobit

EB tresos® E2E Transformer (E2E) documentation

product release 8.8.5

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 of EB tresos E2E Transformer (E2E) documentation

Welcome to the EB tresos E2E Transformer (E2E) (E2EXF) product documentation.

This document provides:

- Chapter 2, “E2EXF release notes“: release notes for the E2EXF modules
- Chapter 3, “E2EXF user's guide“: containing background information and instructions
- Chapter 4, “E2EXF module references“: information about configuration parameters and the application programming interface

2. E2EXF release notes

2.1. Overview

This chapter provides the E2EXF product specific release notes. General release notes that are applicable to all products are provided in the EB tresos AutoCore Generic documentation. Refer to the general release notes in addition to the product release notes documented here.

2.2. Scope of the release

2.2.1. Configuration tool

Your release of EB tresos AutoCore is compatible with the release of the EB tresos Studio configuration tool:

- EB tresos Studio: 28.2.0 b211016-0103

2.2.2. AUTOSAR modules

The following table lists the AUTOSAR modules that are part of this E2EXF release.

Module name	AUTOSAR version and revision	SWS version and revision	Module version	Supplier
E2ESM	4.2.1 []	4.2.1 [0000]	1.1.10	Elektrobit Automotive GmbH
E2EXf	4.2.1 []	4.2.1 [0000]	1.0.37	Elektrobit Automotive GmbH

Table 2.1. Hardware-Independent Modules specified by the AUTOSAR standard

2.2.3. EB (Elektrobit) modules

The following table lists all modules which are part of this release but are not specified by the AUTOSAR standard. These modules include tooling developed by EB or they may hold files shared by all other modules.

Module name	Module version	Supplier
Xfrm	1.0.38	Elektrobit Automotive GmbH

Table 2.2. Modules not specified by the AUTOSAR standard

2.2.4. MCAL modules and EB tresos AutoCore OS

For information about MCAL modules and OS, refer to the respective documentation, which is available as PDF at `$TRESOS_BASE/doc/3.0_EB_tresos_AutoCore_OS` and `$TRESOS_BASE/doc/5.0_MCAL_modules`[1]. It is also available in the online help in EB tresos Studio. Browse to the folders `EB tresos AutoCore OS` and `MCAL modules`.

2.3. Module release notes

2.3.1. E2ESM module release notes

- AUTOSAR R4.2 Rev 1
- AUTOSAR SWS document version: 4.2.1
- Module version: 1.1.10.B520150
- Supplier: Elektrobit Automotive GmbH

2.3.1.1. Change log

This chapter lists the changes between different versions.

Module version 1.1.10
2021-03-05

- Updated preprocessor include guards to be PC-lint compatible

Module version 1.1.9
2020-06-19

[1] `$TRESOS_BASE` is the location at which you installed EB tresos Studio.

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

Module version 1.1.8
2019-10-11

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

Module version 1.1.7
2019-06-14

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

Module version 1.1.6
2018-12-21

- E2ESM_MAX_WINDOW_SIZE is now generated and provided externally by the End-to-End transformer

Module version 1.1.5
2018-10-26

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

Module version 1.1.4
2018-02-01

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

Module version 1.1.3
2017-09-22

- Switch from MISRA-C:2004 to MISRA-C:2012

Module version 1.1.2
2017-08-16

- ASCE2E-500 Fixed known issue: Not possible to integrate Safety Transformer because of incomplete E2ESM

Module version 1.1.1
2016-07-01

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

Module version 1.1.0
2015-09-18

- Implemented improvement for efficient handling of reception window

Module version 1.0.0
2015-06-19

- Initial release.
- Implemented deterministic start-up behavior for E2E State Machine according to Bugzilla #67553

2.3.1.2. New features

- No new features have been added since the last release.

2.3.1.3. EB-specific enhancements

This chapter lists the enhancements provided by the module.

- Deterministic initialization behavior for E2E State Machine

  Description:

  In addition to AUTOSAR release 4.2.1, Bugzilla RfC 67553 is incorporated. That is, a new generic profile status type `E2E_P_NONEWDATA` is introduced in case no new data was received in the actual receive cycle. See also http://www.autosar.org/bugzilla/show_bug.cgi?id=67553 .

- Efficient handling of reception window

  Description:

  Instead of defining explicit window buffers for each inverse E2E Transformer which are referenced in the E2E State Machine objects (see variable ProfileStatusWindow in type definition SWS_E2E_00343), the E2E State Machine specifies a fixed-size array (variable name ProfileStatusWindowArray) with a maximum window size among all inverse transformers. The maximum window size is calculated by the End-to-End Transformer and declared in file E2EXf_Cfg.h.

2.3.1.4. Deviations

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

- Configuration pointer is a pointer to const

  Description:

  In contrast to SWS_E2E_00340 and SWS_E2E_00353, the second argument of `E2E_SMCheck()` and `E2E_SMCheckInit()` is a pointer to const:

  ```
  Std_ReturnType E2E_SMCheck(E2E_PCheckStatusTypeProfileStatus, const E2E_SMConfigType*Config, E2E_SMCheckStateType*State);
  Std_ReturnType E2E_SMCheckInit(E2E_SMCheckStateType*State, const E2E_SMConfigType*Config);
  ```

  See also https://www.autosar.org/bugzilla/show_bug.cgi?id=68903.

  Rationale:

  The E2E configuration is always constant and no write access shall be performed on it.

  Requirements:

  SWS_E2E_00340, SWS_E2E_00353

2.3.1.5. 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 limitations are reported

2.3.1.6. Open-source software

E2ESM does not use open-source software.

2.3.2. E2EXf module release notes

- AUTOSAR R4.2 Rev 1
- AUTOSAR SWS document version: 4.2.1
- Module version: 1.0.37.B520150

- Supplier: Elektrobit Automotive GmbH

2.3.2.1. Change log

This chapter lists the changes between different versions.

Module version 1.0.37
2022-03-09

- Implemented configuration checks in code generator for UpperHeaderBitsToShift, HeaderLength and MaxDeltaCounter parameters of Profile 11
- Fixed generation of uncompilable API of sending transformer if configuration parameter disableEndToEndCheck is enabled

Module version 1.0.36
2021-03-05

- Updated preprocessor include guards to be PC-lint compatible

Module version 1.0.35
2021-01-22

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

Module version 1.0.34
2020-10-23

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

Module version 1.0.33
2020-06-19

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

Module version 1.0.32
2020-01-24

- Implemented configuration checks in code generator for UpperHeaderBitsToShift, HeaderLength and MaxDeltaCounter parameters

Module version 1.0.31
2019-10-11

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

Module version 1.0.30
2019-03-22

- Enhanced support of parameter disableEndToEndCheck to single receivers

Module version 1.0.29
2018-12-21

- Added support for the configuration of initial value of the WaitForFirstData parameter
- Implemented generation of the E2ESM_MAX_WINDOW_SIZE macro

Module version 1.0.28
2018-07-27

- Align memory mapping with safety options according to AUTOSAR 4.3 MetaModel

Module version 1.0.27
2018-06-22

- Extracted safety checker part to module asc_E2EXfCV

Module version 1.0.26
2018-03-16

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

Module version 1.0.25
2018-02-16

- ASCE2EXF-516 Fixed known issue: Generation of incomplete configuration structure causes hard error

Module version 1.0.24
2017-12-15

- Added XfrmIsSafetyTransformer configuration parameter
- Implemented support for configurable type of BufferLength
- Moved profile specific code into E2E profiles

Module version 1.0.23
2017-09-22

- Switch from MISRA-C:2004 to MISRA-C:2012
- Modified initialization sequence

Module version 1.0.22
2017-08-25

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

Module version 1.0.21
2017-07-28

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

Module version 1.0.20
2017-06-30

- Added support of EB tresos E2E Protection Profile 7
- Implemented generic handling of E2E Profile configuration

Module version 1.0.19
2017-03-31

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

Module version 1.0.18
2017-03-03

- ASCE2EXF-325 Fixed known issue: Compile error for configured parameter disableEndToEndCheck and multiple memory partitions

Module version 1.0.17
2017-02-03

- Internal module improvement. This module version update does not affect module functionality
- ASCE2EXF-322 Fixed known issue: Compile error for enabled configuration option disableEndToEnd-Check

Module version 1.0.16
2017-01-05

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

Module version 1.0.15
2016-12-02

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

Module version 1.0.14
2016-11-04

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

Module version 1.0.13
2016-10-07

- Incorporated RfC 75163: Contradicting requirements for no new data on transformer side
- Incorporated RfC 74125: Short guideline on forwarding of legacy COM-based E2E-protectected messages on Ethernet ECUs

Module version 1.0.12
2016-09-09

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

Module version 1.0.11
2016-08-05

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

Module version 1.0.10
2016-07-01

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

Module version 1.0.9
2016-05-25

- Added support for EB tresos E2E Protection Profile 2
- Internal module improvement. This module version update does not affect module functionality

Module version 1.0.8
2016-04-29

- Added support for configuration option EndToEndTransformationComSpecProps.disableEndToEndCheck
- ASCE2EXF-163 Fixed known issue: Soft error for correctly provided data to Rte

Module version 1.0.7
2016-04-01

- Added support for memory partitioning to support partitioned RTE

Module version 1.0.6
2016-02-05

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

Module version 1.0.5
2016-01-15

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

Module version 1.0.4
2015-11-06

- Changed parameter name XfrmTransformationBswModuleEntryRef to XfrmTransformerBswModuleEntryRef (see AUTOSAR RfC #68531)
- Added support for specification of XfrmVariableDataPrototypeInstanceRef (multiple receivers)

Module version 1.0.3
2015-10-09

- Added support of EB tresos E2E Protection Profile 5 and 6

Module version 1.0.2
2015-09-18

- ASCE2EXF-64 Fixed known issue: E2E Transformer for EB tresos E2E Protection Profile 01 always returns with a hard runtime error
- ASCE2EXF-65 Fixed known issue: Initialization of the E2E Transformer via E2EXf_Init results into a soft reset

Module version 1.0.1
2015-08-21

- ASCE2EXF-47 Fixed known issue: E2E Transformer does not generate transformer APIs for data which is (de-)serialized with the Com Based Transformer

Module version 1.0.0
2015-06-19

- Initial release of E2E transformer for EB tresos E2E Protection Profile 1 and 4

2.3.2.2. New features

- No new features have been added since the last release.

2.3.2.3. EB-specific enhancements

This chapter lists the enhancements provided by the module.

- Efficient handling of reception window

  Description:

  Instead of defining explicit window buffers for each inverse E2E transformer which are referenced in the E2E State Machine objects (see variable `ProfileStatusWindow` in type definition SWS_E2E_00343), the E2E State Machine specifies a fixed-size array (variable name `ProfileStatusWindowArray`) with a configurable maximum window size among all inverse transformers.

  Rationale:

  For known use cases, the maximum window size is usually set to at most four which is equal to the size of a pointer variable on common platforms. In this case, applying directly an array of four instead of the pointer to that array is more efficient in both run-time and memory consumption for considered window sizes. Additional information is provided in the release notes of the E2E State Machine.

  Requirement:

  SWS_E2EXf_00021 (partly)

- Configurable data type for BufferLength

  Description:

  Module configuration parameter `XfrmBufferLengthType` enables the user to adjust the data type for parameter `BufferLength` of the transformer APIs.

  Configuring `XfrmBufferLengthType` to `UINT16` sets the data type of parameter `BufferLength` to `uint16`.

  Configuring `XfrmBufferLengthType` to `UINT32` sets the data type of parameter `BufferLength` to `uint32`.

  Rationale:

  This module shall be able to serialize and deserialize data with a size greater than 64 KiB.

- Memory partitioning

  Description:

Module configuration parameter `XfrmOsApplicationRef` enables the user to assign the transformer to a dedicated memory partition.

If parameter `XfrmOsApplicationRef` is enabled and a valid reference to an `OsApplication` is provided, the global variables of the transformer are mapped to the memory partition to which the referenced `OsApplication` belongs.

If parameter `XfrmOsApplicationRef` is disabled, the global variables of the transformer are mapped to the default memory partition.

Rationale:

This enhancement was introduced in order to support the memory partitioning of the RTE.

- Configurable initial value of parameter WaitForFirstData

Description:

Module configuration parameter `E2EWaitForFirstData` enables the user to adjust the initial value of parameter `WaitForFirstData` in profiles P01 and P02.

Configuring `E2EWaitForFirstData` to `FALSE`, the default value, initializes the parameter `WaitForFirstData` in profiles P01 and P02 according to the AUTOSAR 4.2.1 specification, that is with `FALSE`.

Configuring `E2EWaitForFirstData` to `TRUE` initializes the parameter `WaitForFirstData` in profiles P01 and P02 according to the AUTOSAR 4.3.0 (and later) specification, that is with `TRUE`.

Rationale:

This enhancement was introduced to avoid backwards incompatibilities. See also: AR-14174.

2.3.2.4. Deviations

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

- Support of configuration variant pre-compile

Affected AUTOSAR releases:

- R4.2 rev 1

Description:

The E2EXf module supports only configuration variant pre-compile. This is handled in restricting the range of configuration parameter `IMPLEMENTATION_CONFIG_VARIANT` to `VariantPreCompile` which leads to a violation of rule EcucSws_6051.

Requirements:

SWS_E2EXf_00089, SWS_E2EXf_00090, SWS_E2EXf_00126 (partly), SWS_E2EXf_00040 (partly), SWS_E2EXf_00097, SWS_E2EXf_00001 (partly), SWS_E2EXf_00002 (partly), SWS_E2EXf_00144 (partly), SWS_E2EXf_00145 (partly), SWS_E2EXf_00024 (partly), SWS_E2EXf_00035 (partly), SWS_E2EXf_00011, SWS_E2EXf_00096, SWS_E2EXf_00030, SWS_E2EXf_00003 (partly)

- Extended production errors

Description:

Extended production errors are not supported.

Requirements:

ECUC_Xfrm_00016 ECUC_Xfrm_00015 SWS_Xfrm_00070 SWS_Xfrm_00071

- Development errors

Description:

Development errors are not supported. Development error checks are always performed, but development errors are not reported. In case of an error a transformer and an inverted transformer return `E_SAFETY_HARD_RUNTIMEERROR`. The configuration parameter `XfrmDevErrorDetect` is ignored.

Rationale:

Development errors would have to be conform with the highest requested safety standard.

Requirements:

ECUC_Xfrm_00013_Conf, SWS_E2EXf_00137, SWS_E2EXf_00144 (partly), SWS_E2EXf_00145 (partly), SWS_E2EXf_00146, SWS_E2EXf_00149, SWS_E2EXf_00150, SWS_E2EXf_00151, SWS_E2EXf_00152, SWS_E2EXf_00153

- Unsupported protection of SOME/IP Header with PROFILE_01 and PROFILE_02

Description:

PROFILE_01 and PROFILE_02 are intended to be used with the Com transformer as serializing transformer. This implies that the parameters `BufferProperties.headerLength` and `EndToEndTransformationDescription.upperHeaderBitsToShift` are zero.

Rationale:

PROFILE_01 and PROFILE_02 are originally intended to be used with Com transformer and in combination with a Can bus. If a SOME/IP transformer is used above the E2E transformer no bus compatibility to classic COM senders can be achieved. For more information about this issue, see https://www.autosar.org/bugzilla/show_bug.cgi?id=74125.

Requirements:

SWS_E2EXf_00108 (for P01 and P02), SWS_E2EXf_00109 (for P01 and P02), SWS_E2EXf_00155, SWS_E2EXf_00112 (for P01 and P02), SWS_E2EXf_00113 (for P01 and P02)

- File structure differs to AUTOSAR specification

Description:

The code file structure differs from requirement SWS_E2EXf_00003 to support the memory partitioning feature.

In the Elektrobit Automotive GmbH implementation `E2EXf_Cfg.h` does not include header files from the E2E library directly, but via generated compilation units by which transformer APIs are separated into independent memory partitions.

Requirements:

SWS_E2EXf_00003, SWS_E2EXf_00040

- Initializaton routines deviate from AUTOSAR specification

Description:

The API services `E2EXf_Init()` and `E2EXf_DeInit()` are not available. Instead for each partition which has been assigned to a `XfrmImplementationMapping` a dedicated `E2EXf[_<Partition-NameInfix>]_PartitionInit()` respectively `E2EXf[_<PartitionNameInfix>]_Partition-DeInit()` exists.

Dividing initialization routines among partitions allows to use the E2EXf on multi-core systems.

Requirements:

SWS_E2EXf_00024 SWS_E2EXf_00021 SWS_E2EXf_00130 SWS_E2EXf_00133 SWS_E2EXf_00144 SWS_E2EXf_00145 SWS_E2EXf_00035 SWS_E2EXf_00132 SWS_E2EXf_00148 SWS_E2EXf_00146 SWS_E2EXf_00138

- Non-reentrant transformer functions

Description:

The transformer functions `E2EXf_<transformerId>` and `E2EXf_Inv_<transformerId>`, specified as Reentrant, are implemented Non-reentrant.

Rationale:

Each receiver must maintain its own instance of the E2E State Machine (among with the CheckState-Type of the profiles). For more information about this issue, see https://bugzilla.autosar.org/show_bug.-cgi?id=79416.

Requirements:

SWS_E2EXf_00032 SWS_E2EXf_00034

- Violation of VSMD rule Constr_3023

Description:

The attribute apiServicePrefix is mandatory for VSMDs derived from the CDD StMD. The attribute shall not be provided for VSMDs derived from any other StMDs. The rule is based on Constr_3023 from AUTOSAR_TPS_ECUConfiguration.pdf of 4.2.1 Release.

Effected node:

- StMD-Node: `/AUTOSAR/EcucDefs/Xfrm`

Rationale:

The module violates the second part of the constraint, but correctly, because SWS_E2EXf_00156 requires the definition of attribute apiServicePrefix. The discrepancy is handled by AUTOSAR with https://jira.autosar.org/browse/AR-109503 .

Requirements:

SWS_E2EXf_00156

- Violation of VSMD rule EcucSws_2038_2040_ASR41

Description:

If there is a SYMBOLIC-NAME-REFERENCE which points to another module, the rule EcucSws_2038_2040_ASR41, which is based on requirements TPS_ECUC_02038 and TPS_ECUC_02040 from AUTOSAR_TPS_ECUConfiguration.pdf of 4.1 Release, always creates a violation.

Effected nodes:

- VSMD-Node: `Xfrm/XfrmImplementationMapping/XfrmDemEventParameterRefs/XFRM_E_MALFORMED_MESSAGE`
- VSMD-Node: `Xfrm/XfrmImplementationMapping/XfrmOsApplicationRef`

Rationale:

Violation occurs due to an invalid behavior within the EB tresos Studio. No useful workaround available. Rule EcucSws_2038_2040_ASR41 shall be ignored.

Requirements:

ECUC_Xfrm_00016

2.3.2.5. 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 limitations are reported

2.3.2.6. Open-source software

E2EXf does not use open-source software.

2.3.3. Xfrm module release notes

- AUTOSAR R4.0 Rev 3
- AUTOSAR SWS document version: 0.0.0
- Module version: 1.0.38.B520150
- Supplier: Elektrobit Automotive GmbH

2.3.3.1. Change log

This chapter lists the changes between different versions.

Module version 1.0.38
2022-03-09

- Fixed invalid consideration of configuration container EndToEndTransformationComSpecProps for sending transformers

Module version 1.0.37
2022-01-28

- ASCXFRM-439, ASCCOMXF-639 Fixed known issue: Code generation for multiple receivers results in an error

Module version 1.0.36
2021-11-10

- ASCXFRM-434, ASCCOMXF-628 Fixed known issue: Safety Com transformer forwards undersized buffer length to the E2EXf

Module version 1.0.35
2021-10-08

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

Module version 1.0.34
2021-09-17

- Improved error message for the invalid ImplementationDataType reference of the container SwDataDefProps
- Added support for SomeIpXf_ExtractProtocolHeaderFields API

Module version 1.0.33
2021-04-09

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

Module version 1.0.32
2021-03-05

- Updated preprocessor include guards to be PC-lint compatible

Module version 1.0.31
2021-02-12

- Implemented support for data types with identifier (tag/length/value)

Module version 1.0.30
2020-07-31

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

Module version 1.0.29
2020-05-22

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

Module version 1.0.28
2020-01-24

- ASCXFRM-365 Fixed known issue: Variable Size Arrays are serialized to the wrong array type

Module version 1.0.27
2019-10-11

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

Module version 1.0.26
2019-03-22

- Enhanced support of parameter disableEndToEndCheck to single receivers

Module version 1.0.25
2019-02-15

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

Module version 1.0.24
2018-10-26

- Prevent transformers from generating invalid BSWMD.arxml when configuration of XfrmImplementation-Mapping is empty

Module version 1.0.23
2018-07-27

- Align memory mapping with safety options according to AUTOSAR 4.3 MetaModel

Module version 1.0.22
2018-06-22

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

Module version 1.0.21
2018-05-25

- ASCXFRM-309 Fixed known issue: Compile error in generated SomeIpXf_Api_gen.h

Module version 1.0.20
2018-03-16

- Updated the compatibility check interface

Module version 1.0.19
2017-12-15

- Implemented support for configurable type of BufferLength
- Added XfrmIsSafetyTransformer configuration parameter
- Updated the supported datatypes to AUTOSAR release 4.2.1

Module version 1.0.18
2017-09-22

- Add safe transformer condition check

Module version 1.0.17
2017-07-28

- Implemented usage of ApplicationDataType of (outermost) SwComponentType

Module version 1.0.16
2017-06-02

- Added support for Safe ComXf

Module version 1.0.15
2017-05-05

- Improved gathering of computational methods
- Added support for variable size array profile of type fully flexible

Module version 1.0.14
2017-03-31

- Incorporated Bugzilla RfC 69896: Execution of Transformer chain in case of unqueued communication when no data is available
- Incorporated Bugzilla RfC 68623: Insufficient specification of autonomous error response

Module version 1.0.13
2017-03-03

- ASCXFRM-150 Fixed known issue: Compile error occurs when ImplementationDataType of the outermost CompositionSwComponent differs from atomic software component for client/server communication

Module version 1.0.12
2017-02-03

- Enable Xfrm users to gather computational methods from the abstracted data type model by providing a reference to SwDataDefProps

Module version 1.0.11
2017-01-05

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

Module version 1.0.10
2016-12-02

- ASCXFRM-144 Fixed known issue: Compile error occurs when ImplementationDataType of the outermost CompositionSwComponent differs from atomic software component for sender/receiver communication

Module version 1.0.9
2016-11-04

- Incorporated Bugzilla RfC 70485: Missing configuration parameter XfrmVersionInfoApi

Module version 1.0.8
2016-10-07

- Implemented usage of ImplementationDataType of (outermost) SwComponentType

Module version 1.0.7
2016-09-09

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

Module version 1.0.6
2016-07-01

- ASCXFRM-82 Fixed known issue: Generic SwBaseTypes when referenced by ImplementationDataType lead to incorrect generated configuration items

Module version 1.0.5
2016-05-25

- Provided error message for usage of unsupported basic data types

Module version 1.0.4
2016-04-29

- Added support for generic SwBaseType

Module version 1.0.3
2016-04-01

- Incorporated Bugzilla RfC 67775: SOME/IP Transformer uses wrong length of length field (extensible fixed size arrays)

Module version 1.0.2
2016-02-05

- Addded support of ApplicationDataType
- Incorporated Bugzilla RfC 68085: Clarification issues regarding the modeling of Variable-Size Array Data Type

Module version 1.0.1
2015-11-06

- Changed parameter name XfrmTransformationBswModuleEntryRef to XfrmTransformerBswModuleEntryRef (see AUTOSAR RfC #68531)
- Added support for specification of XfrmVariableDataPrototypeInstanceRef (multiple receivers)

Module version 1.0.0
2015-10-09

- Initial release of Transformer library

2.3.3.2. New features

- No new features have been added since the last release.

2.3.3.3. EB-specific enhancements

This chapter lists the enhancements provided by the module.

- This module provides no EB-specific enhancements.

2.3.3.4. Deviations

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

- For this module no deviations are known.

2.3.3.5. 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.

- For this module no limitations are known.

2.3.3.6. Open-source software

Xfrm does not use open-source software.

3. E2EXF user's guide

3.1. Overview

This user's guide describes the E2EXf module. From this user's guide you learn the basic functionality of the E2EXf. You also learn which related modules are necessary to configure the E2EXf module. The E2EXf module reference provides further information on how to configure the E2EXf itself.

Note that this user's guide is intended for readers who have good knowledge of AUTOSAR and about the purpose of the E2EXf. The information provided here helps you to integrate the E2EXf in your AUTOSAR project.

- Section 3.2, “Background information” provides an overview of the basic functionality of the E2EXf.
- Section 3.3, “Configuring E2EXf” provides information on related modules that are needed in order to configure the E2EXf.
- Section 3.4, “E2EXf integration notes” provides notes for the integration of the E2EXf module into your project.
- For details on how to configure the E2EXf itself, see the parameter descriptions that are provided in Chapter 4, “E2EXF module references“.

3.2. Background information

The general concept of data transformation and large data transfer is described in the EB tresos AutoCore Generic documentation. If you are not familiar with this topic, see the section Data transformation that is part of the Concepts chapter in the generic documentation.

3.2.1. Functional overview

The E2EXf is a safety transformer that implements data transformation according to the E2EXf AUTOSAR specification (see [1]).

It is used together with the modules Rte, ComXf and SomeIpXf to add protection information to the serialized data stream for the following communication paradigm:

Sender-receiver communication

The E2EXf provides APIs to add protection information to the results of a serializing transformer (e.g. ComXf or SomeIpXf) at the sender side, and to check communication errors cyclically by using this information at the receiver side. The E2EXf API functions are called by the Rte.

Information about client-server communication

A server can have multiple clients, so a dataID can not be uniquely assigned. Therefore, the `E2EXf` does not support adding protection information for the communication paradigm client-server communication, as described in the Limitations section of the E2EXf AUTOSAR specification (see [1]).

Information about crc and counters

The `E2EXf` provides APIs to add protection information (crc and counter value) using one of the profiles `E2EP01`, `E2EP02`, `E2EP04`, `E2EP05`, `E2EP06`, `E2EP07`, `E2EPRN` or `E2EPJLR`.

3.3. Configuring E2EXf

To configure the `E2EXf` module, add the module to your project using EB tresos Studio. Parameter descriptions are provided to guide the configuration. You find these in the module references section of this document. You also find these parameter descriptions in the online help of EB tresos Studio.

NOTE 

Non-editable parameter `XfrmIsSafetyTransformer`

The parameter `XfrmIsSafetyTransformer` is disabled for all `XfrmImplementationMappings` which are part of the `E2EXf` module configuration. The rationale is that a manual overwrite of the classification of an `E2EXf` transformation (contrary to `SomeIpXf` and `ComXf`) shall be avoided.

To use the `E2EXf` module, you must configure additional modules as outlined below:

`Rte` module

The `E2EXf` module provides API functions called from the `Rte` module in ACG8 RTE (Rte).

After you import the AUTOSAR system or software component description with the project in EB tresos Studio, execute the unattended wizard **Calculate Service Needs(SvcAs_Trigger)** to fill the module configuration with the required `XfrmImplementationMappings`. Additional parameters to enable or disable the Development Error Tracer (`Det`) or defensive programming are provided in the tab **General** of the module configuration.

Library modules

The `E2EXf` module requires the following library modules to add or check protection information (see [2]):

- AUTOSAR E2E State Machine (`E2ESM`, called from the `E2EXf` module)

  Add this module to your project in EB tresos Studio. There are no parameters which have to be configued.

- AUTOSAR E2E Profile (`E2EPxx`, called from the `E2ESM` module)

  Add the `E2EPxx` module with `xx` which specifies the profile ID (e.g., `E2EP01`, `E2EP04`) to your project in EB tresos Studio. There are no parameters which have to be configured.

- Generic data types for E2E Profiles (`E2E`, called from the `E2EPxx` module)

Add this module to your project in EB tresos Studio. There are no parameters which have to be configued.

- Cyclic Redundancy Check (CRC) routines (`SCrc`, called from the `E2EPxx` module)

  Add this module to your project in EB tresos Studio. There are no parameters which have to be configued.

These library modules do not provide any configuration parameters.

3.4. E2EXf integration notes

You find general integration information in the EB tresos AutoCore Generic documentation.

In addition, you find module-specific information about exclusive areas, production errors and memory mapping in the module-specific integration notes. You find the module-specific integration notes in the module references chapter of this document. See Chapter 4, “E2EXF module references“ sub-section `Integration notes` in each module.

3.4.1. End-to-End check

The attribute `disableEndToEndCheck` is configured in container `EndToEndTransformationISignalComSpecProps` within the system model configuration. If `disableEndToEndCheck` is configured to true then the `E2EXf` module on receiver side does not perform any validation of the data. If `disableEndToEndCheck` is configured to false then the validation of the data is performed on the receiver side.

3.4.2. Memory partitioning

To avoid any negative impact of the `E2EXf` module on the overall safety integrity level for the application all its global variables (e.g. internal state variables of type `E2EXf_P01CheckStatesType` for the individual transformers of profile P01) are placed in a memory section with safety integrity level ASIL-D. On the other hand the memory partitions to which these variables are assigned depend on the value configured for parameter `XfrmOsApplicationRef`. This should be set to the OS-Application of the SWC which sends or receives the particular PDU. Finally, the overall safety integrity level is determined by the memory section with the lowest safety integrity level.

4. E2EXF module references

4.1. Overview

This chapter provides module references for the E2EXF product modules. These include a detailed description of all configuration parameters. Furthermore this chapter lists the application programming interface with all data types, constants and functions.

The content of the sections is sorted alphabetically according the EB tresos AutoCore Generic module names.

For further information on the functional behavior of these modules, refer to the chapter E2EXF user's guide.

4.1.1. Notation in EB module references

EB notation may differ from the AUTOSAR standard notation in the software specification documents (SWS). This section describes the notation of *default value* and *range* fields in the EB module references.

4.1.1.1. Default value of configuration parameters

If there is no default value specified for a parameter, the default value field is omitted to prevent ambiguity with parameters that have `--` as default values.

Example: The parameter `BswMCompuConstText` of the `BswM` module of EB tresos AutoCore Generic 8 Mode Management has no default value field, therefore it is omitted.

4.1.1.2. Range information of configuration parameters

The range of a configuration parameter contains an upper and a lower boundary. However, in special cases the range of allowed values can be computed by means of an XPath function that is evaluated at configuration time. An XPath function can either be a standard `xpath:<function>()` or a custom `cxpath:<function>()` function. The range of a configuration parameter may be computed based on other configuration parameters that are referenced from the XPath function. For more information on custom XPath functions, see section *Custom XPath Functions API* of the EB tresos Studio developer's guide.

Example: The parameter `BswMCompuConstText` of the `BswM` module of EB tresos AutoCore Generic 8 Mode Management has the custom XPath function `cxpath:getCompuMethodsVT()` in the range field which provides the allowed values.

4.2. E2ESM

4.2.1. Configuration parameters

Containers included		
Container name	**Multiplicity**	**Description**
CommonPublishedInformation	1..1	**Label:** Common Published Information Common container, aggregated by all modules. It contains published information about vendor and versions.
PublishedInformation	1..1	**Label:** EB Published Information Additional published parameters not covered by CommonPublishedInformation container.

4.2.1.1. CommonPublishedInformation

Parameters included	
Parameter name	**Multiplicity**
ArMajorVersion	1..1
ArMinorVersion	1..1
ArPatchVersion	1..1
SwMajorVersion	1..1
SwMinorVersion	1..1
SwPatchVersion	1..1
ModuleId	1..1
VendorId	1..1
VendorApiInfix	1..1
Release	1..1

Parameter Name	ArMajorVersion
Label	AUTOSAR Major Version
Description	Major version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1

Type	INTEGER_LABEL
Default value	4
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**ArMinorVersion**
Label	AUTOSAR Minor Version
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	2
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**ArPatchVersion**
Label	AUTOSAR Patch Version
Description	Patch level version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**SwMajorVersion**
Label	Software Major Version
Description	Major version number of the vendor specific implementation of the module.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**SwMinorVersion**

Label	Software Minor Version
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**SwPatchVersion**
Label	Software Patch Version
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	10
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**ModuleId**
Label	Numeric Module ID
Description	Module ID of this module from Module List
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**VendorId**
Label	Vendor ID
Description	Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list
Multiplicity	1..1
Type	INTEGER_LABEL

Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**VendorApiInfix**
Multiplicity	1..1
Type	STRING_LABEL

Parameter Name	**Release**
Label	Release Information
Multiplicity	1..1
Type	STRING_LABEL
Default value	
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

4.2.1.2. PublishedInformation

Parameters included	
Parameter name	**Multiplicity**
PbcfgMSupport	1..1

Parameter Name	**PbcfgMSupport**
Label	PbcfgM support
Description	Specifies whether or not the E2ESM can use the PbcfgM module for post-build support.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

4.2.2. Application programming interface (API)

4.2.2.1. Type definitions

4.2.2.1.1. E2E_SMCheckStateType

Purpose	Definition of E2E State Machine check state type.	
Type	struct	
Members	`uint8 ProfileStatusWindowArray`	Array in which the ProfileStatus of the last E2E-checks are stored. The array size is fixed with E2ESM_MAX_WINDOW_SIZE.
	`uint8 WindowTopIndex`	Index in the array at which the next ProfileStatus is to be written.
	`uint8 OkCount`	Count of checks in which ProfileStatus equal to E2E_P_OK was determined within the last WindowSize checks.
	`uint8 ErrorCount`	Count of checks in which ProfileStatus equal to E2E_P_ERROR was determined within the last WindowSize checks.
	`E2E_SMStateType SMState`	The current state in the state machine.
Description	State of the protection of a communication channel.	

4.2.2.1.2. E2E_SMConfigType

Purpose	Definition of E2E State Machine communication channel configuration type.	
Type	struct	
Members	`uint8 WindowSize`	Size of the monitoring window for the state machine.
	`uint8 MinOkStateInit`	Minimal number of checks in which ProfileStatus equal to E2E_P_OK was determined within the last WindowSize checks (for the state E2E_SM_INIT) required to change to state E2E_SM_VALID.
	`uint8 MaxErrorStateInit`	Maximal number of checks in which ProfileStatus equal to E2E_P_ERROR was determined within the last WindowSize checks (for the state E2E_SM_INIT).
	`uint8 MinOkStateValid`	Minimal number of checks in which ProfileStatus equal to E2E_P_OK was deter-

		mined within the last WindowSize checks (for the state E2E_SM_VALID) required to keep in state E2E_SM_VALID.
	`uint8 MaxErrorStateValid`	Maximal number of checks in which ProfileStatus equal to E2E_P_ERROR was determined, within the last WindowSize checks (for the state E2E_SM_VALID).
	`uint8 MinOkStateInvalid`	Minimum number of checks in which ProfileStatus equal to E2E_P_OK was determined within the last WindowSize checks (for the state E2E_SM_INVALID) required to change to state E2E_SM_VALID.
	`uint8 MaxErrorStateInvalid`	Maximal number of checks in which ProfileStatus equal to E2E_P_ERROR was determined within the last WindowSize checks (for the state E2E_SM_INVALID).
Description	Configuration of a communication channel for exchanging Data.	

4.2.2.1.3. E2E_SMStateType

Purpose	Definition of E2E State Machine state type.
Type	`uint8`
Description	Status of the communication channel exchanging the data. If the status is OK, then the data may be used.

4.2.2.2. Macro constants

4.2.2.2.1. E2ESM_AR_RELEASE_MAJOR_VERSION

Purpose	AUTOSAR release major version.
Value	4U

4.2.2.2.2. E2ESM_AR_RELEASE_MINOR_VERSION

Purpose	AUTOSAR release minor version.
Value	2U

4.2.2.2.3. E2ESM_AR_RELEASE_REVISION_VERSION

Purpose	AUTOSAR release revision version.
Value	1U

4.2.2.2.4. E2ESM_SW_MAJOR_VERSION

Purpose	AUTOSAR module major version.
Value	1U

4.2.2.2.5. E2ESM_SW_MINOR_VERSION

Purpose	AUTOSAR module minor version.
Value	1U

4.2.2.2.6. E2ESM_SW_PATCH_VERSION

Purpose	AUTOSAR module patch version.
Value	10U

4.2.2.2.7. E2ESM_VENDOR_ID

Purpose	AUTOSAR vendor identification: Elektrobit Automotive GmbH.
Value	1U

4.2.2.2.8. E2E_SM_DEINIT

Purpose	State before E2E_SMCheckInit() is invoked.
Value	1U

4.2.2.2.9. E2E_SM_INIT

Purpose	There has been yet no communication since the initialization.
Value	3U

4.2.2.2.10. E2E_SM_INVALID

Purpose	There has been some communication since startup, but not enough to switch to VALID.
Value	4U

4.2.2.2.11. E2E_SM_NODATA

Purpose	No data is available since the initialization. This means there is no communication and there is no E2E-valid default value available.
Value	2U

4.2.2.2.12. E2E_SM_VALID

Purpose	Communication functioning properly according to E2E.
Value	0U

4.2.2.3. Functions

4.2.2.3.1. E2E_SMCheck

Purpose	Check the received Data using the E2E State Machine.	
Synopsis	`Std_ReturnType E2E_SMCheck ( E2E_PCheckStatusType ProfileStatus , const E2E_SMConfigType * ConfigPtr , E2E_SMCheckStateType * StatePtr );`	
Service ID	0x30	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different check states	
Parameters (in)	`ProfileStatus`	Profile-independent status of the reception on one single Data in one cycle
	`ConfigPtr`	Pointer to static configuration.
Parameters (in,out)	`StatePtr`	Pointer to port/data communication state.
Return Value	Function execution success status	
	`E2E_E_INPUTERR_NULL`	At least one pointer parameter is a NULL pointer.

	E2E_E_INPUTERR_WRONG	At least one input parameter is erroneous.
	E2E_E_INTERR	At least one input parameter is erroneous.
	E2E_E_WRONGSTATE	Function executed in wrong state.
	E2E_E_OK	Function completed successfully.
Description	Checks the communication channel. It determines if the data can be used for safety-related application, based on history of checks performed by a corresponding E2E_P0XCheck() function.	

4.2.2.3.2. E2E_SMCheckInit

Purpose	Initializes the E2E state machine.	
Synopsis	`Std_ReturnType` **`E2E_SMCheckInit`** `( E2E_SMCheckStateType * StatePtr , const E2E_SMConfigType * ConfigPtr );`	
Service ID	0x31	
Sync/Async	Synchronous	
Reentrancy	Reentrant for different check states	
Parameters (in)	ConfigPtr	Pointer to static configuration.
Parameters (in,out)	StatePtr	Pointer to port/data communication state.
Return Value	Function execution success status	
	E2E_E_INPUTERR_NULL	At least one pointer parameter is a NULL pointer.
	E2E_E_INPUTERR_WRONG	At least one input parameter is erroneous.
	E2E_E_OK	Function completed successfully.

4.2.3. Integration notes

4.2.3.1. Exclusive areas

Exclusive areas are not used by the E2ESM module.

4.2.3.2. Production errors

Production errors are not reported by the E2ESM module.

4.2.3.3. Memory mapping

General information about memory mapping is provided in the EB tresos AutoCore Generic documentation. Refer to the section `Memory mapping and compiler abstraction` in the `Integration notes` section for details.

The following table provides the list of sections that may be mapped for this module:

Memory section
CODE

4.2.3.4. Integration requirements

WARNING **Integration requirements list is not exhaustive**

The following list of integration requirements helps you to integrate your product. However, this list is not exhaustive. You also require information from the user guide, release notes, and EB tresos AutoCore known issues to successfully integrate your product.

Integration requirements are not listed for the E2ESM module.

4.3. E2EXf

4.3.1. Configuration parameters

Containers included		
Container name	**Multiplicity**	**Description**
CommonPublishedInformation	1..1	**Label:** Common Published Information Common container, aggregated by all modules. It contains published information about vendor and versions.
PublishedInformation	1..1	**Label:** EB Published Information Additional published parameters not covered by CommonPublishedInformation container.
XfrmGeneral	1..1	Contains the general configuration parameters of the module.
XfrmImplementationMapping	1..n	For each transformer (TransformationTechnology) in a transformer chain (DataTransformation) which is applied to an

Containers included		
		ISignal it is necessary to specify the BswModuleEntry which implements it. This is the container to hold these mappings.

Parameters included	
Parameter name	**Multiplicity**
IMPLEMENTATION_CONFIG_VARIANT	1..1

Parameter Name	IMPLEMENTATION_CONFIG_VARIANT
Label	Config Variant
Multiplicity	1..1
Type	ENUMERATION
Default value	VariantPreCompile
Range	VariantPreCompile

4.3.1.1. CommonPublishedInformation

Parameters included	
Parameter name	**Multiplicity**
ArMajorVersion	1..1
ArMinorVersion	1..1
ArPatchVersion	1..1
SwMajorVersion	1..1
SwMinorVersion	1..1
SwPatchVersion	1..1
ModuleId	1..1
VendorId	1..1
Release	1..1

Parameter Name	ArMajorVersion
Label	AUTOSAR Major Version
Description	Major version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL

Default value	4
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**ArMinorVersion**
Label	AUTOSAR Minor Version
Description	Minor version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	2
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**ArPatchVersion**
Label	AUTOSAR Patch Version
Description	Patch level version number of AUTOSAR specification on which the appropriate implementation is based on.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**SwMajorVersion**
Label	Software Major Version
Description	Major version number of the vendor specific implementation of the module.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**SwMinorVersion**

Label	Software Minor Version
Description	Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	0
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**SwPatchVersion**
Label	Software Patch Version
Description	Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	37
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**ModuleId**
Label	Numeric Module ID
Description	Module ID of this module from Module List
Multiplicity	1..1
Type	INTEGER_LABEL
Default value	176
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**VendorId**
Label	Vendor ID
Description	Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list
Multiplicity	1..1
Type	INTEGER_LABEL

Default value	1
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

Parameter Name	**Release**
Label	Release Information
Multiplicity	1..1
Type	STRING_LABEL
Default value	
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

4.3.1.2. PublishedInformation

Parameters included	
Parameter name	**Multiplicity**
PbcfgMSupport	1..1

Parameter Name	**PbcfgMSupport**
Label	PbcfgM support
Description	Specifies whether or not the E2EXf can use the PbcfgM module for post-build support.
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Configuration class	**PublishedInformation:**
Origin	Elektrobit Automotive GmbH

4.3.1.3. XfrmGeneral

Parameters included	
Parameter name	**Multiplicity**
XfrmBufferLengthType	1..1

Parameters included	
XfrmDevErrorDetect	1..1
XfrmVersionInfoApi	1..1
E2EWaitForFirstData	1..1

Parameter Name	XfrmBufferLengthType	
Description	Specifies the data type of parameter BufferLength for transformer APIs.	
Multiplicity	1..1	
Type	ENUMERATION	
Default value	UINT16	
Range	UINT16 UINT32	
Configuration class	**VariantPreCompile:**	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	XfrmDevErrorDetect	
Description	*Development errors are not supported.* All checks are performed, but no development errors will be reported.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	false	
Configuration class	**VariantPreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	XfrmVersionInfoApi	
Description	Activate/Deactivates the version information API.	
Multiplicity	1..1	
Type	BOOLEAN	
Default value	false	
Configuration class	**VariantPreCompile:**	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	E2EWaitForFirstData
Description	Specifies the initial value of the parameter WaitForFirstData in profiles P01 and P02.

	▶ FALSE: The parameter WaitForFirstData is initialized with value FALSE. (Default - specified before Autosar version 4.3.0, for backward compatibility) ▶ TRUE: The parameter WaitForFirstData is initialized with value TRUE. (specified since Autosar version 4.3.0)
Multiplicity	1..1
Type	BOOLEAN
Default value	false
Configuration class	**VariantPreCompile:** VariantPreCompile
Origin	Elektrobit Automotive GmbH

4.3.1.4. XfrmImplementationMapping

Containers included		
Container name	**Multiplicity**	**Description**
XfrmVariableDataPrototypeInstanceRef	1..1	Instance reference to a VariableDataPrototype in case a dedicated transformer BswModuleEntry is required per VariableDataPrototype access.
XfrmDemEventParameterRefs	1..1	Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameters DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.
XfrmSignal	1..1	Reference to the signal in the system description that transports the transformed data.

Parameters included	
Parameter name	**Multiplicity**
XfrmTransformerBswModuleEntryRef	0..1
XfrmInvTransformerBswModuleEntryRef	0..1
XfrmTransformationTechnologyRef	1..1
XfrmIsSafetyTransformer	0..1
XfrmOsApplicationRef	0..1

Parameter Name	**XfrmTransformerBswModuleEntryRef**

Description	Reference to the BswModuleEntry which implements the referenced transformer on the sending/calling side.	
Multiplicity	0..1	
Type	FOREIGN-REFERENCE	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	**XfrmInvTransformerBswModuleEntryRef**	
Description	Reference to the BswModuleEntry which implements the referenced inverse transformer on the receiving/called side.	
Multiplicity	0..1	
Type	FOREIGN-REFERENCE	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**XfrmTransformationTechnologyRef**	
Description	Reference to the TransformationTechnology in the DataTransformation of the system description for which the implementation (BswModuleEntry) shall be mapped.	
Multiplicity	1..1	
Type	FOREIGN-REFERENCE	
Configuration class	**VariantPreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

Parameter Name	**XfrmIsSafetyTransformer**	
Description	E2EXf is always considered as a safety transformer. Specifies if the Transformer shall be considered as a safety Transformer.	
Multiplicity	0..1	
Type	BOOLEAN	
Default value	false	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	Elektrobit Automotive GmbH	

Parameter Name	**XfrmOsApplicationRef**
Description	Reference to an Os application to which the BSW belongs.

	► `Enabled`: Maps global variables of transformer or inverted transformer function to dedicated memory partition. ► `Disabled`: No dedicated memory partition assigned.
Multiplicity	0..1
Type	REFERENCE
Configuration class	**PreCompile:** VariantPreCompile
Origin	Elektrobit Automotive GmbH

4.3.1.5. XfrmVariableDataPrototypeInstanceRef

Parameters included	
Parameter name	**Multiplicity**
TARGET	1..1
CONTEXT	2..2

Parameter Name	**TARGET**
Description	Target of the instance reference to a VariableDataPrototype. The context of the instance reference is given by the list of Context References (CONTEXT) below.
Multiplicity	1..1
Type	REFERENCE
Configuration class	**VariantPreCompile:** VariantPreCompile
Origin	AUTOSAR_ECUC

Parameter Name	**CONTEXT**
Description	List of ordered context references of the instance reference to: 1. SwComponentPrototype 2. PortPrototype The target of the instance reference is given with the VariableDataPrototype reference (TARGET) above.
Multiplicity	2..2
Type	REFERENCE
Range	SW-COMPONENT-PROTOTYPE

	PORT-PROTOTYPE*	
Configuration class	**PreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.3.1.6. XfrmDemEventParameterRefs

Parameters included	
Parameter name	**Multiplicity**
XFRM_E_MALFORMED_MESSAGE	0..1

Parameter Name	**XFRM_E_MALFORMED_MESSAGE**	
Description	*The functionality related to this parameter is not supported by the current implementation.* Reference to configured DEM event to report if malformed messages were received by the transformer. Reference to configured DEM event to report if malformed messages were received by the transformer.	
Multiplicity	0..1	
Type	SYMBOLIC-NAME-REFERENCE	
Configuration class	**VariantPreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.3.1.7. XfrmSignal

Containers included		
Container name	**Multiplicity**	**Description**
XfrmSignalChoice	1..1	Choice whether an ISignal or an ISignalGroup shall be referenced.

4.3.1.8. XfrmSignalChoice

Containers included		
Container name	**Multiplicity**	**Description**
XfrmISignalGroupRefChoice	1..1	Reference to the ISignalGroup in the system description that transports the transformed data.

Containers included		
XfrmISignalRefChoice	1..1	Reference to the ISignal in the system description that transports the transformed data.

4.3.1.9. XfrmISignalGroupRefChoice

Parameters included	
Parameter name	**Multiplicity**
XfrmISignalGroupRef	1..1

Parameter Name	XfrmISignalGroupRef	
Description	Reference to the ISignalGroup in the system description that transports the transformed data.	
Multiplicity	1..1	
Type	FOREIGN-REFERENCE	
Configuration class	**VariantPreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.3.1.10. XfrmISignalRefChoice

Parameters included	
Parameter name	**Multiplicity**
XfrmISignalRef	1..1

Parameter Name	XfrmISignalRef	
Description	Reference to the ISignal in the system description that transports the transformed data.	
Multiplicity	1..1	
Type	FOREIGN-REFERENCE	
Configuration class	**VariantPreCompile:**	VariantPreCompile
Origin	AUTOSAR_ECUC	

4.3.2. Application programming interface (API)

4.3.2.1. Macro constants

4.3.2.1.1. E2EXF_SID_GETVERSIONINFO

Purpose	API Service ID for E2EXf_GetVersioninfo().
Value	((uint8)0x00U)

4.3.2.1.2. E2EXF_SID_INV_TRANSFORMER

Purpose	API Service ID for E2EXf_Inv_transformerId() communication.
Value	((uint8)0x04U)

4.3.2.1.3. E2EXF_SID_TRANSFORMER

Purpose	API Service ID for E2EXf_transformerId() communication.
Value	((uint8)0x03U)

4.3.2.2. Functions

4.3.2.2.1. E2EXf_GetVersionInfo

Purpose	Get version information of the E2EXf module.	
Synopsis	`void E2EXf_GetVersionInfo ( Std_VersionInfoType * VersionInfo );`	
Service ID	0x00	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (out)	`VersionInfo`	Pointer to where to store the version information of this module.
Description	This service returns the version information of this module. The version information includes: ► Module Id ► Vendor Id ► Vendor specific version numbers	

4.3.2.2.2. E2EXf_InPlace_Inv_transformerId

Purpose	This function checks the received data. using the in-place inverse transformation. If the data can be used by the caller, then the function returns E_OK.	
Synopsis	`uint8 E2EXf_InPlace_Inv_transformerId ( uint8 * BufferPtr , uint16 * BufferLengthPtr , uint16 InputBufferLength );`	
Service ID	0x04	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	`InputBufferLength`	This argument holds the length of the E2E transformer's input data (in the inputBuffer argument). If executeDespiteDataUnavailability is set to true and the transformer is executed without valid input data, the length will be equal to 0.
Parameters (in,out)	`BufferPtr`	This is the buffer where the E2E transformer places its output data. With the E2E transformer configured for in-place transformation, it also contains its input data. With the E2E transformer using in-place transformation and having a headerLength different from 0, the output data of the previous transformer begin at position headerLength. It is the buffer allocated by the RTE, where the transformed data has to be stored by the transformer.
Parameters (out)	`BufferLengthPtr`	Used length of the buffer
Return Value	Function execution success status. The high nibble represents the state of the E2E state machine, the low nibble represents the status of the last E2E check. For the following return codes, please see SWS Transformer General:	
	`0x00`	(E_OK) This means VALID_OK
	`0x01`	(E_SAFETY_VALID_REP)
	`0x02`	(E_SAFETY_VALID_SEQ)
	`0x03`	(E_SAFETY_VALID_ERR)
	`0x05`	(E_SAFETY_VALID_NND)
	`0x20`	(E_SAFETY_NODATA_OK)
	`0x21`	(E_SAFETY_NODATA_REP)
	`0x22`	(E_SAFETY_NODATA_SEQ)

	0x23	(E_SAFETY_NODATA_ERR)
	0x25	(E_SAFETY_NODATA_NND)
	0x30	(E_SAFETY_INIT_OK)
	0x31	(E_SAFETY_INIT_REP)
	0x32	(E_SAFETY_INIT_SEQ)
	0x33	(E_SAFETY_INIT_ERR)
	0x35	(E_SAFETY_INIT_NND)
	0x40	(E_SAFETY_INVALID_OK)
	0x41	(E_SAFETY_INVALID_REP)
	0x42	(E_SAFETY_INVALID_SEQ)
	0x43	(E_SAFETY_INVALID_ERR)
	0x45	(E_SAFETY_INVALID_NND)
	0x77	(E_SAFETY_SOFT_RUNTIMEERROR): A runtime error occured, safety properties could not be checked (state or status cannot be determined) but non-protected output data could be produced nonetheless.
	0xFF	(E_SAFETY_HARD_RUNTIMEERROR): A runtime error occured, safety properties could not be checked and no output data could be produced.

4.3.2.2.3. E2EXf_InPlace_transformerId

Purpose	Protects the array/buffer to be transmitted, using the in-place transformation.	
Synopsis	`uint8 E2EXf_InPlace_transformerId ( uint8 * BufferPtr , uint16 * BufferLengthPtr , uint16 InputBufferLength );`	
Service ID	0x03	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	InputBufferLength	This argument holds the length of the E2E transformer's input data (in the inputBuffer argument). If executeDespiteDataUnavailability is set to true and the transformer is executed without valid input data, the length will be equal to 0.

Parameters (in,out)	`BufferPtr`	This is the buffer where the E2E transformer places its output data. With the E2E transformer configured for in-place transformation, it also contains its input data. With the E2E transformer using in-place transformation and having a headerLength different from 0, the output data of the previous transformer begin at position headerLength. It is the buffer allocated by the RTE, where the transformed data has to be stored by the transformer.
Parameters (out)	`BufferLengthPtr`	Used length of the buffer
Return Value	Function execution success status	
	`0x00`	(E_OK): Function performed successfully.
	`0x77`	(E_SAFETY_SOFT_RUNTIMEERROR): A runtime error occurred, safety properties could not be checked (state or status cannot be determined) but non-protected output data could be produced nonetheless.
	`0xFF`	(E_SAFETY_HARD_RUNTIMEERROR): A runtime error occurred, safety properties could not be checked and no output data could be produced.

4.3.2.2.4. E2EXf_OutOfPlace_Inv_transformerId

Purpose	This function checks the received data. using the out-of-place inverse transformation. If the data can be used by the caller, then the function returns E_OK.	
Synopsis	`uint8` **`E2EXf_OutOfPlace_Inv_transformerId`** `( uint8 * BufferPtr , uint16 * BufferLengthPtr , const uint8 * InputBufferPtr , uint16 InputBufferLength );`	
Service ID	0x04	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	`InputBufferPtr`	This argument holds the input data for the transformer. If executeDespiteDataUnavailability is set to true, Rte will hand over a NULL pointer to the transformer.

	`InputBufferLength`	This argument holds the length of the E2E transformer's input data (in the inputBuffer argument). If executeDespiteDataUnavailability is set to true and the transformer is executed without valid input data, the length will be equal to 0.
Parameters (out)	`BufferPtr`	This is the buffer where the E2E transformer places its output data. It is the buffer allocated by the RTE, where the transformed data has to be stored by the transformer.
	`BufferLengthPtr`	Used length of the buffer
Return Value	Function execution success status. The high nibble represents the state of the E2E state machine, the low nibble represents the status of the last E2E check. For the following return codes, please see SWS Transformer General:	
	`0x00`	(E_OK) This means VALID_OK
	`0x01`	(E_SAFETY_VALID_REP)
	`0x02`	(E_SAFETY_VALID_SEQ)
	`0x03`	(E_SAFETY_VALID_ERR)
	`0x05`	(E_SAFETY_VALID_NND)
	`0x20`	(E_SAFETY_NODATA_OK)
	`0x21`	(E_SAFETY_NODATA_REP)
	`0x22`	(E_SAFETY_NODATA_SEQ)
	`0x23`	(E_SAFETY_NODATA_ERR)
	`0x25`	(E_SAFETY_NODATA_NND)
	`0x30`	(E_SAFETY_INIT_OK)
	`0x31`	(E_SAFETY_INIT_REP)
	`0x32`	(E_SAFETY_INIT_SEQ)
	`0x33`	(E_SAFETY_INIT_ERR)
	`0x35`	(E_SAFETY_INIT_NND)
	`0x40`	(E_SAFETY_INVALID_OK)
	`0x41`	(E_SAFETY_INVALID_REP)
	`0x42`	(E_SAFETY_INVALID_SEQ)
	`0x43`	(E_SAFETY_INVALID_ERR)
	`0x45`	(E_SAFETY_INVALID_NND)

	0x77	(E_SAFETY_SOFT_RUNTIMEERROR): A runtime error occurred, safety properties could not be checked (state or status cannot be determined) but non-protected output data could be produced nonetheless.
	0xFF	(E_SAFETY_HARD_RUNTIMEERROR): A runtime error occurred, safety properties could not be checked and no output data could be produced.

4.3.2.2.5. E2EXf_OutOfPlace_transformerId

Purpose	Protects the array/buffer to be transmitted, using the out-of-place transformation.	
Synopsis	`uint8 E2EXf_OutOfPlace_transformerId ( uint8 * BufferPtr , uint16 * BufferLengthPtr , const uint8 * InputBufferPtr , uint16 InputBufferLength );`	
Service ID	0x03	
Sync/Async	Synchronous	
Reentrancy	Reentrant	
Parameters (in)	InputBufferPtr	This argument holds the input data for the transformer. If executeDespiteDataUnavailability is set to true, Rte will hand over a NULL pointer to the transformer.
	InputBufferLength	This argument holds the length of the E2E transformer's input data (in the inputBuffer argument). If executeDespiteDataUnavailability is set to true and the transformer is executed without valid input data, the length will be equal to 0.
Parameters (out)	BufferPtr	This is the buffer where the E2E transformer places its output data. It is the buffer allocated by the RTE, where the transformed data has to be stored by the transformer.
	BufferLengthPtr	Used length of the buffer
Return Value	Function execution success status	
	0x00	(E_OK): Function performed successfully.

	0x77	(E_SAFETY_SOFT_RUNTIMEERROR): A runtime error occured, safety properties could not be checked (state or status cannot be determined) but non-protected output data could be produced nonetheless.
	0xFF	(E_SAFETY_HARD_RUNTIMEERROR): A runtime error occured, safety properties could not be checked and no output data could be produced.

4.3.2.2.6. E2EXf_PartitionDeInit

Purpose	Deinitializes the state of the E2E Transformer.
Synopsis	`void` **`E2EXf_PartitionDeInit`** `( void );`
Service ID	0x02
Sync/Async	Synchronous
Reentrancy	Re-entrant

4.3.2.2.7. E2EXf_PartitionInit

Purpose	Initializes the state of the E2E Transformer.	
Synopsis	`uint8` **`E2EXf_PartitionInit`** `( void );`	
Service ID	0x01	
Sync/Async	Synchronous	
Reentrancy	Re-entrant	
Return Value		
Description	This function initializes the E2E library state structures, which is done by calling init-functions from all configured partitions.	

4.3.3. Integration notes

4.3.3.1. Exclusive areas

Exclusive areas are not used by the `E2EXf` module.

4.3.3.2. Production errors

Production errors are not reported by the `E2EXf` module.

4.3.3.3. Memory mapping

General information about memory mapping is provided in the EB tresos AutoCore Generic documentation. Refer to the section `Memory mapping and compiler abstraction` in the `Integration notes` section for details.

The following table provides the list of sections that may be mapped for this module:

Memory section
CODE
VAR_FAST_INIT_UNSPECIFIED
CONFIG_DATA_UNSPECIFIED
VAR_INIT_8
VAR_NO_INIT_UNSPECIFIED
CONST_32
DEFAULTPARTITION_VAR_INIT_ASIL_D_8
DEFAULTPARTITION_VAR_CLEARED_ASIL_D_UNSPECIFIED
<PartitionNameInfix>_VAR_INIT_ASIL_D_8
<PartitionNameInfix>_VAR_CLEARED_ASIL_D_UNSPECIFIED

4.3.3.4. Integration requirements

WARNING 

Integration requirements list is not exhaustive

The following list of integration requirements helps you to integrate your product. However, this list is not exhaustive. You also require information from the user's guide, release notes, and EB tresos AutoCore known issues to successfully integrate your product.

4.3.3.4.1. E2EXf.EB.IntReq.CyclicCallByRte01

Description	Cyclical call of Function E2EXf_-"transformerId"
Rationale	This represents use case requirement UC_E2EXf_00007 which is considered as integration requirement.

4.3.3.4.2. E2EXf.EB.IntReq.CyclicCallByRte02

Description	Cyclical call of Function E2EXf_Inv_-"transformerId"
Rationale	This represents use case requirement UC_E2EXf_00008 which is considered as integration requirement.

4.3.3.4.3. E2EXf.EB_IntReq.InitRoutines

Description	E2EXf_Init() and E2EXf_DeInit() do not exist. Each partition shall invoke its individual de-/initialization routine namely E2EXf[_<PartitionNameInfix>]_PartitionInit respectively E2EXf[_<PartitionNameInfix>]_PartitionDeInit.
Rationale	Different transformations might be mapped to different partitions. Every partition must provide their own initialization routine in order to support multi-core systems.

5. Bibliography

Bibliography

[1] *AUTOSAR Specification of Module E2E Transformer*, Issue AUTOSAR Release 4.2.1, Publisher: AUTOSAR

[2] *AUTOSAR Specification of SW-C End-to-End Communication Protection Library*, Issue AUTOSAR Release 4.2.1, Publisher: AUTOSAR