MICROSAR E2E

Technical Reference

Version 1.02.00

Authors	Michael Goß
Status	Released

Document Information

History

Author	Date	Version	Remarks
Michael Goß	2015-07-03	1.00.00	Initial version
Michael Goß	2015-10-21	1.01.00	Support of JLR E2E profile
Michael Goß	2016-11-25	1.02.00	Support of E2E profile 7

Reference Documents

No.	Source	Title	Version
[1]	AUTOSAR	AUTOSAR_SWS_E2ELibrary.pdf	V4.2.1
[2]	AUTOSAR	AUTOSAR_SWS_E2ELibrary.pdf	V4.3.0
[3]	AUTOSAR	AUTOSAR_TR_BSWModuleList.pdf	V4.2.1

Scope of the Document

This technical reference describes the general use of the E2E library basis software. The E2E Library was developed according to ISO 26262 for use in safety-related items. There are no aspects which are controller specific.

Caution
We have configured the programs in accordance with your specifications in the questionnaire. Whereas the programs do support other configurations than the one specified in your questionnaire, Vector´s release of the programs delivered to your company is expressly restricted to the configuration you have specified in the questionnaire.

Contents

Illustrations

Tables

1 Component History

The component history gives an overview over the important milestones that are supported in the different versions of the component.

Component Version	New Features
1.00.00	Initial creation of E2E Library documentation summarizing all supported E2E profiles.
1.01.00	Documentation was updated according to additional E2E profile for JLR
1.02.00	Documentation was updated according to additional E2E profile 7

Table 1-1 Component history

2 Introduction

This document describes the functionality and API of the AUTOSAR BSW module E2E as specified in [1].

Supported AUTOSAR Release*:	4	
Supported Configuration Variants:	-	
Vendor ID:	E2E_VENDOR_ID	30 decimal (= Vector-Informatik, according to HIS)
Module ID:	E2E_MODULE_ID	207 decimal (according to ref. [3])

* For the detailed functional specification please also refer to the corresponding AUTOSAR SWS.

E2E Library provides mechanisms to protect safety-related data exchange at runtime against the effects of faults within the communication link.

To provide appropriate solution addressing flexibility and standardization, AUTOSAR specifies a set of flexible E2E profiles that implement a combination of E2E protection mechanisms, i.e. Profile 1, 2, 4, 5, 6 and 7. Additionally, an E2E profile for JLR is supported, which is based on AUTOSAR Profile 1.Each specified E2E profile has a fixed behavior, but it has some configuration options by function parameters (e.g. the location of the CRC in relation to the data, which are to be protected). This document illustrates the functional principle of E2E without getting too deep into details about any particular E2E profile. For information about the used E2E profile (e.g. data layout, CRC computation), refer to [1] and [2].

The E2E protection allows the following:

> It protects the safety-related data elements to be sent over the RTE by attaching control data (E2E Header)

> It verifies the safety-related data elements received from the RTE using the control data (E2E Header)

> It indicates that received safety-related data elements are faulty, so that the caller of E2E library can respond in a proper way

E2E Library provides a state machine to signalize the state of communication link over several message cycles.

The E2E Profile check-functions verify data in one cycle. In contrary, the state machine builds up a state out of several results of check-functions within a reception window, which is then provided to the consumer (RTE/SWC/COM).

E2E State Machine should be interpreted more like an aggregator of several check-results than as a real state machine.

2.1 Architecture Overview

The following figure shows where the E2E is located in the AUTOSAR architecture.

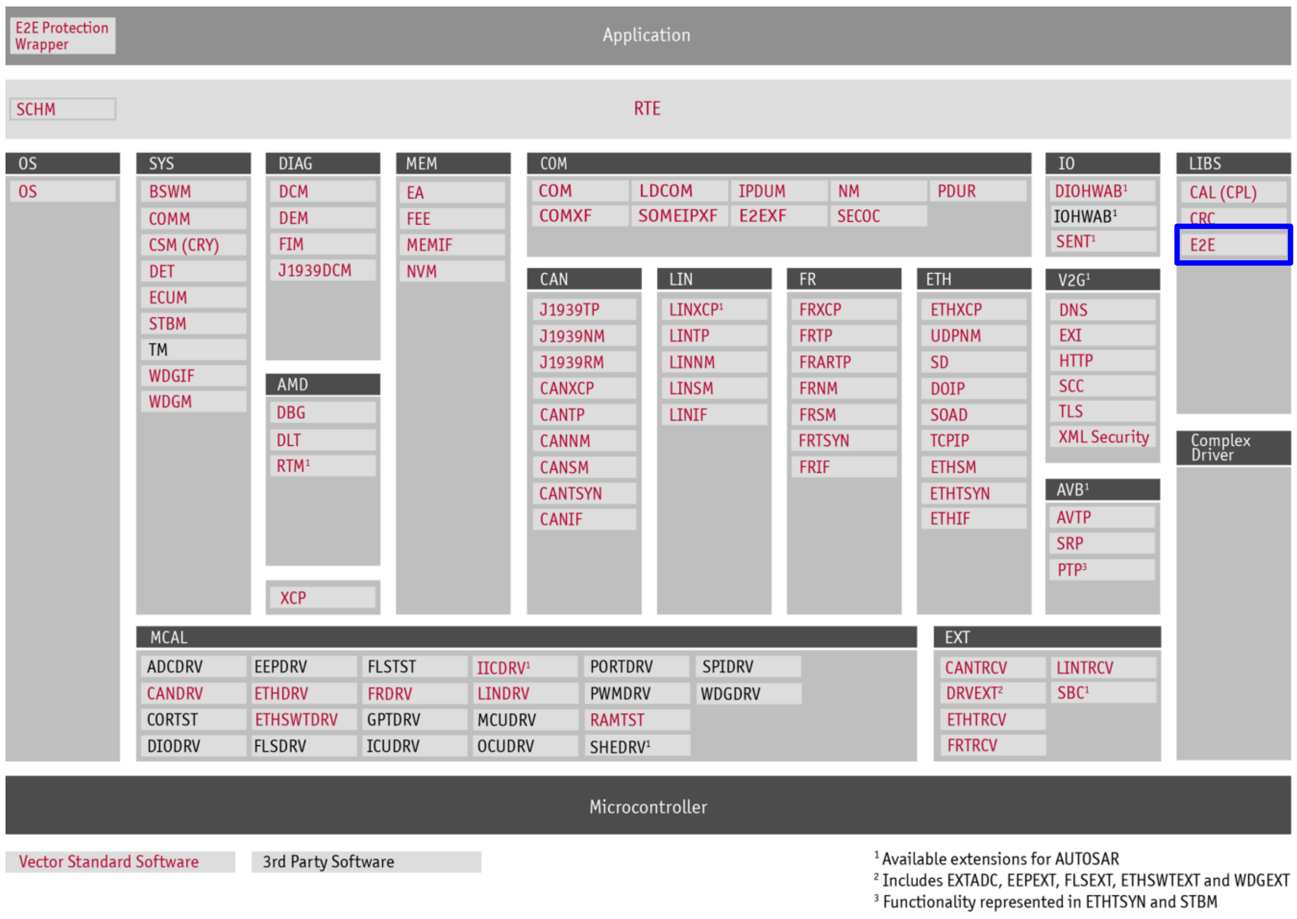

Figure 2-1 AUTOSAR 4.2 Architecture Overview

The next figure shows the interfaces to adjacent modules of the E2E. These interfaces are described in chapter 5.

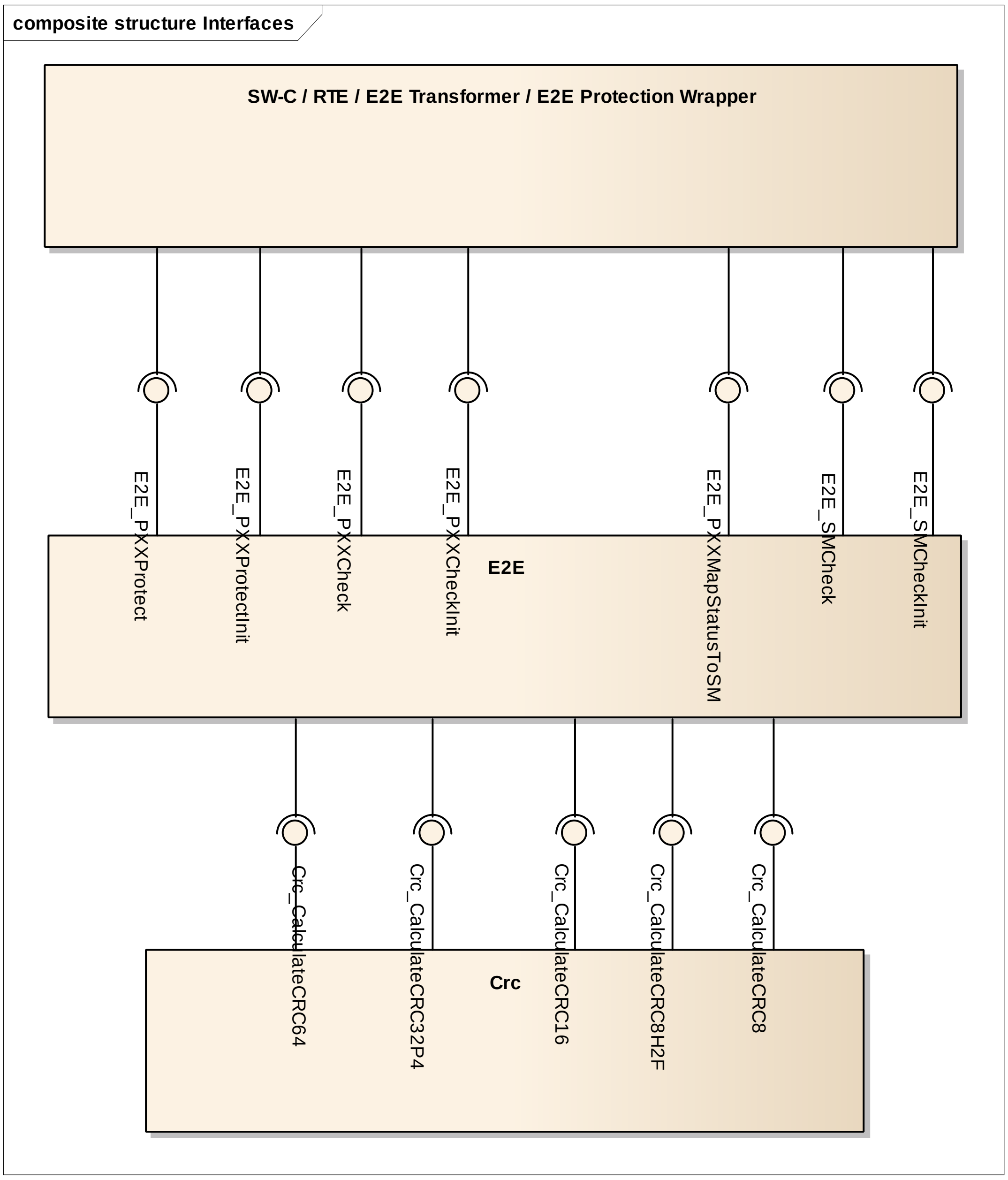

Figure 2-2 Interfaces to adjacent modules of E2E component

PXX is used as placeholder for services, which are implemented in each Profile. For example E2E_**PXX**Protect() develops to the following six E2E_**P01**Protect(), E2E_**P02**Protect(), E2E_**P04**Protect(), E2E_**P05**Protect(), E2E_**P06**Protect(), E2E_**P07**Protect() and E2E_**PJLR**Protect().

3 Functional Description

3.1 Features

The features listed in the following tables cover the complete functionality specified for the E2E.

The AUTOSAR standard functionality is specified in [1], the corresponding features are listed in the table

> Table 3-1 Supported AUTOSAR standard conform features

Vector Informatik provides further E2E functionality beyond the AUTOSAR standard. The corresponding features are listed in the table

> Table 3-2 Features provided beyond the AUTOSAR standard

The following features specified in [1] are supported:

Supported AUTOSAR Standard Conform Features
E2E Protection mechanisms via Profile 1, 2, 4, 5, 6 and 7
State machine for communication status within reception window

Table 3-1 Supported AUTOSAR standard conform features

The following features are provided beyond the AUTOSAR standard:

Features Provided Beyond the AUTOSAR Standard
E2E Protection mechanism via Profile JLR

Table 3-2 Features provided beyond the AUTOSAR standard

3.2 E2E Communication Protection

This chapter gives an overview over E2E communication protection and provides a short summary of the AUTOSAR architecture for E2E communication protection. Please note that this chapter is not intended to provide a full description (or even a specification) of the AUTOSAR architecture for E2E communication protection.

3.2.1 Communication Faults

To ensure freedom from interference by software partitioning, ISO 26262 takes the following faults into account:

> Loss of peer-to-peer communication

> Unintended message repetition due to the same message being sent again unintentionally

> Message loss during transmission

> Insertion of messages due to receiver unintentionally receiving an additional message, which is interpreted to have correct source and destination addresses

> Re-sequencing due to the order of data being changed during transmission, i.e., the data is not received in the same order as it was sent
> Message corruption due to one or more data bits in the message begin changed during transmission
> Message delay due to the message being received correctly, but not in time
> Blocking access to the data bus due to a faulty node not adhering to the expected patterns of use and making excessive demands for service, thereby reducing its availability to other nodes, e.g. while wrongly waiting for non-existent data
> Constant transmission of messages by a faulty node, thereby compromising the operation of the entire bus

These communication faults may arise due to (systematic) software faults, (random) hardware faults and transient faults caused by external influences.

3.2.2 Fault Model

The following communication faults can be addressed by the E2E communication protection:

> Repetition: The same message is received more than once
> Deletion: The message or parts of it have been removed from the communication stream
> Insertion: An additional message or parts of it have been inserted into the communication stream
> Incorrect sequence: Messages of a communication stream are received in an incorrect order
> Corruption: A message or parts of it are modified
> Timing faults (delay): The timing constraints of a message are violated
> Addressing faults: A message is sent to the wrong destination
> Inconsistency: Communicating nodes have different views of the network status of data being transferred
> Masquerading: The design of a received message with non-authentic content appears authentic, as if sent by the correct sender

The specific features of the E2E communication protection depend on the implementation of the protection mechanisms.

3.2.3 Protection mechanisms

The following mechanisms are provided by E2E profiles. Note that combination and implementation of protection mechanisms differ in each E2E Profile. Table 3-3 summarizes the possible mechanisms. For details about the protection mechanisms in any specific E2E Profile, refer to [1].

E2E Mechanism	Description
Counter	Representing sequence number, which is incremented on every send request. Counter is either transmitted implicitly by including in the CRC calculation or it is transmitted explicitly as well as covered by CRC calculation.
Timeout monitoring	Timeout is determined by E2E Library by means of evaluation of the Counter, by a non-blocking read at the receiver. Timeout is reported by E2E Library to the caller by means of the status flags in E2E_PXXCheckStatusType.
Data ID	Unique number which is transmitted either implicitly (only included in CRC computation) or explicitly and covered by CRC calculation.
Length	Length of data being transmitted is sent explicitly by some E2E profiles.
CRC	The CRC routines are provided by CRC library. Entire E2E Header and data to be transmitted is included in CRC calculation.

Table 3-3 Provided E2E protection mechanisms

3.2.4 Detected Communication Faults

The E2E mechanisms can detect the following faults or effects of faults:

E2E Mechanism	Detected communication faults
Counter	Repetition, Loss, Insertion, Incorrect sequence, Blocking
Transmission on a regular basis and timeout monitoring using E2E-Library	Loss, Delay, Blocking
Data ID + CRC	Masquerade and incorrect addressing, Insertion
CRC	Corruption, Asymmetric information

Table 3-4 Detected communication faults

3.3 Initialization

The caller of E2E Library has to provide configuration parameters according to type definitions, which are defined in section 5.1. E2E Protection and check services do require state parameters which are to be initialized before first protect/check cycle. The state machine also requests a state parameter which requires initialization. Therefore E2E provides several initialization services, which are responsible for resetting state parameters.

> E2E_PXXProtectInit has to be called before first protection

> E2E_PXXCheckInit has to be called before first check

> E2E_SMCheckInit has to be called to initialized the state machine

3.4 States

E2E state machine indicates the communication status over several protect/check cycles.

According to state machine configuration which is passed to E2E_SMCheck service via input parameter, communication link is trustworthy as long as state machine is in state E2E_SM_VALID. Received data is ok to use then.

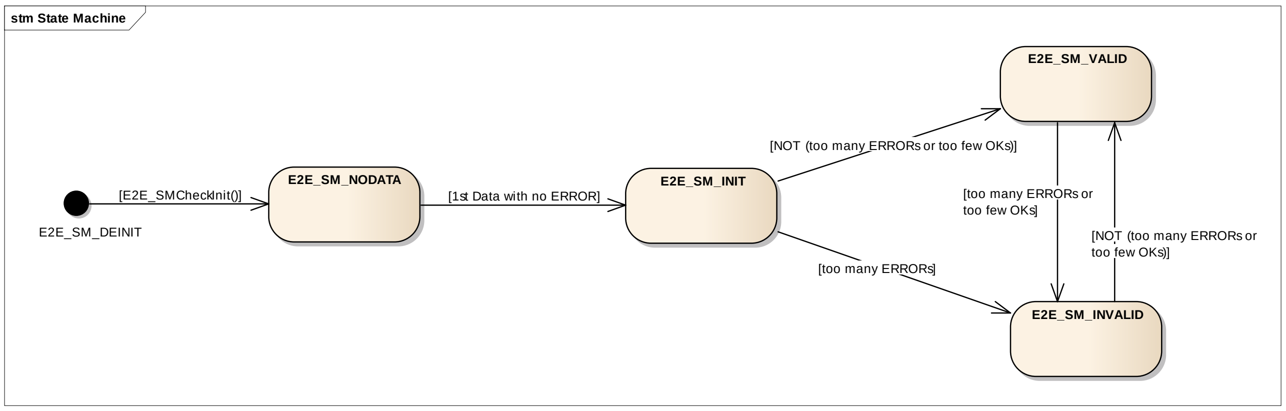

Figure 3-1 E2E State Machine

E2E Profile services are stateless.

3.5 Main Functions

E2E does not contain any scheduled services such as a main function.

3.6 Error Handling

E2E does not support error reporting via DET or DEM modules. Errors are reported synchronously upon call of services via return values and status within state parameters.

4 Integration

This chapter gives necessary information for the integration of the MICROSAR E2E into an application environment of an ECU.

4.1 Scope of Delivery

The delivery of the E2E contains the files which are described in the chapters 4.1.1 and 4.1.2:

4.1.1 Static Files

File Name	Description
E2E.c	This is the common source file of E2E Library.
E2E.h	This is the common header file of E2E Library.
E2E_PXX.c	This is the specific source file for E2E Profile PXX.
E2E_PXX.h	This is the specific header file for E2E Profile PXX.
E2E_SM.c	This is the source file for E2E State Machine.
E2E_SM.h	This is the header file for E2E State Machine.

Table 4-1 Static files

PXX is a placeholder for all supported profiles. Upon delivery of E2E Library only one particular profile is supported.

It is important to note that only the header file E2E_PXX.h has to be included to the caller of E2E Library. See chapter 4.2 for include structure of E2E Library.

4.1.2 Dynamic Files

E2E Library does not contain any generated files.

4.2 Include Structure

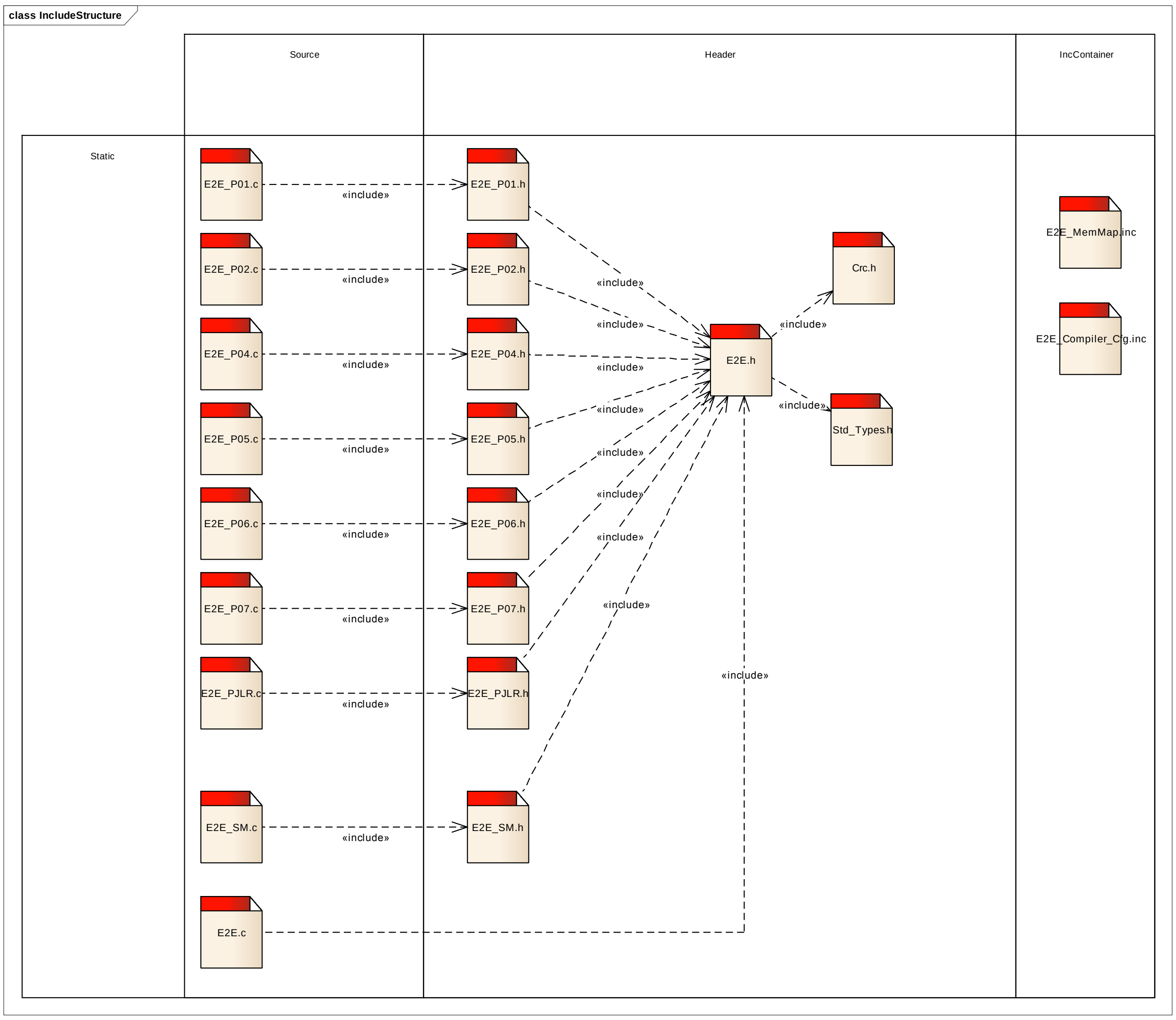

Figure 4-1 Include structure containing all possible E2E profiles

According to Figure 4-1 Include structure containing all possible E2E profiles it is sufficient to include the E2E_PXX.h header file, e.g. E2E_P01.h if E2E Profile 1 is used and to include E2E_SM.h if the state machine is used.

4.3 Make

The content of make sub-package has to be adapted according to the E2E Profile, which is used in a delivery. Therefore set parameter `E2E_USE_PROFILEX` to `1` in E2E_cfg.mak file. For example, if E2E Profile 1 is used:

Setting Profile 1 in E2E_cfg.mak

```
##########################################################################
# Use PROFIL1
#
# Possible Values:
# 0 - No PROFIL1 used
# 1 - PROFIL1 used
##########################################################################
E2E_USE_PROFIL1 = 1
```

Table 4-2 Example for setting E2E Profile 1 in E2E_cfg.mak

4.4 Critical Sections

E2E Library does not contain critical sections.

4.5 Invocation of Protect and Check service

- The service to protect `E2E_PXXProtect` (e.g. `E2E_P01Protect`) shall be called exactly once before sending the data.
- The service to check `E2E_PXXCheck` (e.g. `E2E_P01Check`) shall be called exactly once for reception of data.

5 API Description

For an interfaces overview please see

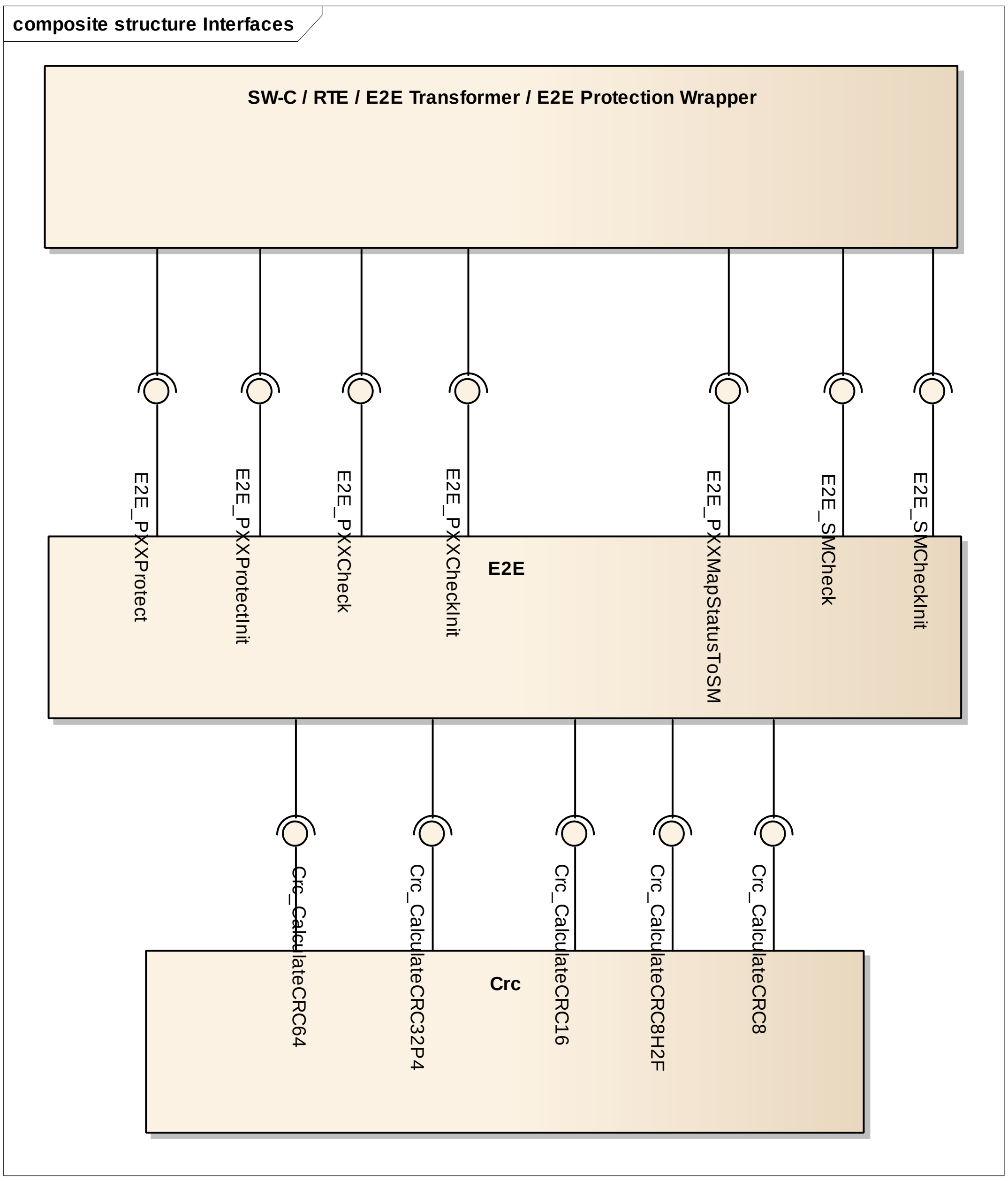

Figure 2-2 Interfaces to adjacent modules of E2E component.

5.1 Type Definitions

The types defined by the E2E are described in this chapter. Content of structure types varies from profile to profile. Hence, profile types are described in general.

Type Name	C-Type	Description
E2E_PXXConfigType	Structure	Configuration of transmitted data. In detail, configuration

Type Name	C-Type	Description
		of E2E Header including protection mechanisms, which are transmitted explicitly. For each transmitted data, there is an instance of this structure.
E2E_PXXProtectState Type	Structure	State of the sender for a data protected with E2E Profile X. In all E2E profiles the Counter is contained in ProtectState structure.
E2E_PXXCheckStateT ype	Structure	State of the receiver for single data protected with E2E Profile X.
E2E_PXXCheckStatus Type	Enumeration	Status of the receiver for single data in one cycle, protected with E2E Profile X. E2E_PXXCheckStatusType is part of E2E_PXXCheckStateType.
E2E_PCheckStatusTy pe	Enumeration	Profile-independent status of the receiver for single data in once cycle. Result and status of most recent check is mapped (by E2E_PXXMapStatusToSM service) to this type, which is compatible to E2E state machine.
E2E_SMConfigType	Structure	Configuration of a communication channel for exchanging data. Number of considered messages (window size) can be configured, which is used to indicate state of communication link over several transmission cycles.
E2E_SMCheckStateTy pe	Structure	State of the protection of a communication channel. State machine state is contained in this type.
E2E_SMStateType	Enumeration	Status of the communication channel exchanging the data. If the status is E2E_SM_VALID, then the data may be used.

Table 5-1 Type definitions

5.2 Services provided by E2E

The services provided by E2E library are described in this chapter. Signature of services may vary from profile to profile. Thus, profile specific parameters are marked appropriately.

5.2.1 E2E_PXXProtect

Prototype	
`Std_ReturnType` **`E2E_PXXProtect`** `( E2E_PXXConfigType* ConfigPtr, E2E_PXXProtectStateType* StatePtr, uint8* DataPtr, uint16 Length)`	
Parameter	
`ConfigPtr`	Pointer to profile configuration
`StatePtr`	Pointer to communication state of sender
`DataPtr`	Pointer to data to be protected
`Length`	Length of data in bytes **(only supported in Profile 4, 5 and 6)**. Type of Length parameter in **Profile 7** is uint32.

Return code	
`Std_ReturnType`	> E2E_E_INPUTERR_NULL: At least one pointer parameter is NULL > E2E_E_INPUTERR_WRONG: One parameter is erroneous > E2E_E_INTERR: Internal library error > E2E_E_OK: Protection successful
Functional Description	
Protects the array/buffer to be transmitted using the E2E Profile X. This includes checksum calculation, handling of counter and Data ID. Depending on the used profile, also data length can be included in E2E Header.	
Particularities and Limitations	
> Identical ConfigPtr should be used for both protection on sender side and check on receiver side. > Only relevant for Profile 5: Value of parameter Length must match length, which is configured in ConfigPtr.	
Expected Caller Context	
> There is no specific caller. E2E can be called from anywhere.	

Table 5-2 E2E_PXXProtect

5.2.2 E2E_PXXProtectInit

Prototype	
`Std_ReturnType` **`E2E_PXXProtectInit`** `(E2E_PXXProtectStateType* StatePtr)`	
Parameter	
`StatePtr`	Pointer to communication state of sender
Return code	
`Std_ReturnType`	> E2E_E_INPUTERR_NULL: Pointer parameter is NULL > E2E_E_OK: Initialization successful
Functional Description	
Initializes the protection state by resetting the counter.	
Particularities and Limitations	
> None	
Expected Caller Context	
> There is no specific caller. E2E can be called from anywhere.	

Table 5-3 E2E_PXXProtectInit

5.2.3 E2E_PXXCheck

Prototype	
`Std_ReturnType` **`E2E_PXXCheck`** `( E2E_PXXConfigType* ConfigPtr, E2E_PXXProtectStateType* StatePtr, uint8* DataPtr, uint16 Length)`	
Parameter	
`ConfigPtr`	Pointer to profile configuration
`StatePtr`	Pointer to communication state of sender
`DataPtr`	Pointer to data to be protected
`Length`	Length of data in bytes **(only supported in Profile 4, 5 and 6)**. Type of Length parameter in **Profile 7** is uint32.
Return code	
`Std_ReturnType`	> E2E_E_INPUTERR_NULL: At least one pointer parameter is NULL > E2E_E_INPUTERR_WRONG: One parameter is erroneous > E2E_E_INTERR: Internal library error > E2E_E_OK: Check successful
Functional Description	
Checks the received data using the E2E Profile X. This includes CRC calculation, handling of the counter and Data ID. Depending on the used profile, also data length can be used for checking received data.	
Particularities and Limitations	
> Identical ConfigPtr should be used for both protection on sender side and checking on receiver side. > Only relevant for Profile 5: Value of parameter Length must match length, which is configured in ConfigPtr.	
Expected Caller Context	
> There is no specific caller. E2E can be called from anywhere.	

Table 5-4 E2E_PXXCheck

5.2.4 E2E_PXXCheckInit

Prototype	
`Std_ReturnType` **`E2E_PXXCheckInit`** `(E2E_PXXCheckStateType* StatePtr)`	
Parameter	
`StatePtr`	Pointer to communication state of sender
Return code	
`Std_ReturnType`	> E2E_E_INPUTERR_NULL: Input pointer parameter is NULL > E2E_E_OK: Initialization successful
Functional Description	
Initializes the check state by resetting the parameters contained in StatePtr.	

Particularities and Limitations
> None
Expected Caller Context
> There is no specific caller. E2E can be called from anywhere.

Table 5-5 E2E_PXXCheckInit

5.2.5 E2E_PXXMapStatusToSM

Prototype	
`E2E_PCheckStatusType` **`E2E_PXXMapStatusToSM`** `( Std_ReturnType CheckReturn, E2E_PXXCheckStatusType* Status, Boolean profileBehavior)`	
Parameter	
`CheckReturn`	Return value of E2E_PXXCheck service
`Status`	Status of received and checked data in single cycle. Status is part of E2E_PXXCheckStateType.
`profileBehavior`	> TRUE: profile behavior as introduced in ASR4.2 > FALSE: profile behavior as before ASR4.2 **This parameter is only supported in Profile 1, 2 and JLR.**
Return code	
`E2E_PCheckStatusType`	Standard state value to be used in E2E Library state machine.
Functional Description	
Maps the check status of Profile X to a generic check status, which can be used by E2E state machine check function. The E2E Profile X delivers a more fine-grained status than it is necessary for E2E state machine.	
Particularities and Limitations	
> None	
Expected Caller Context	
> There is no specific caller. E2E can be called from anywhere.	

Table 5-6 E2E_PXXMapStatusToSM

5.2.6 E2E_SMCheck

Prototype	
`Std_ReturnType` **`E2E_SMCheck`** `( E2E_PCheckStatusType ProfileStatus, E2E_SMConfigType* ConfigPtr, E2E_SMCheckStateType* StatePtr)`	
Parameter	
`ProfileStatus`	Profile independent status of the reception on one single data in one cycle. This is the mapped return value of E2E_PXXMapStatusToSM service.
`ConfigPtr`	Pointer to static configuration of E2E State Machine
`StatePtr`	Pointer to state of the communication channel

Return code	
`Std_ReturnType`	> E2E_E_INPUTERR_NULL: At least one pointer parameter is NULL > E2E_E_INPUTERR_WRONG: One parameter is erroneous > E2E_E_INTERR: Internal library error > E2E_E_OK: State machine check successful > E2E_E_WRONGSTATE: Invalid ProfileStatus
Functional Description	
Processes E2E State Machine and 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_PXXCheck function.	
Particularities and Limitations	
> None	
Expected Caller Context	
> There is no specific caller. E2E can be called from anywhere.	

Table 5-7 E2E_SMCheck

5.2.7 E2E_SMCheckInit

Prototype	
`Std_ReturnType` **`E2E_SMCheckInit`** `(E2E_SMCheckStateType* StatePtr, E2E_SMConfigType* ConfigPtr)`	
Parameter	
`StatePtr`	Pointer to state of the communication channel
`ConfigPtr`	Pointer to static configuration of E2E State Machine
Return code	
`Std_ReturnType`	> E2E_E_INPUTERR_NULL: At least one pointer parameter is NULL > E2E_E_OK: Initialization successful
Functional Description	
Initializes state of the communication channel by resetting all parameters in state structure.	
Particularities and Limitations	
> None	
Expected Caller Context	
> There is no specific caller. E2E can be called from anywhere.	

Table 5-8 E2E_SMCheck

5.3 Services used by E2E

In the following table services provided by other components, which are used by the E2E are listed. For details about prototype and functionality refer to the documentation of the providing component.

Component	API
CRC	> Crc_CalculateCRC8 > Crc_CalculateCRC8H2F > Crc_CalculateCRC16 > Crc_CalculateCRC32P4 > Crc_CalculateCRC64

Table 5-9 Services used by the E2E

5.4 Callback Functions

E2E Library does not provide any callback functions.

5.5 Configurable Interfaces

E2E Library does not provide any configurable interfaces.

6 Configuration

E2E Library is non-generated, deterministic software code, where all inputs and settings are passed by function parameters.

For both protection on sender side and checking on receiver side a state structure has to be provided to the services E2E_PXXProtect (sender) and E2E_PXXCheck (receiver). Additionally, both sender and receiver require a configuration structure of type E2E_PXXConfigType, which specifies details about the E2E Header and its containing protection mechanisms. The content of configuration structure is static and needs to be specified by the caller of E2E Library.

Caution
Static configuration for data (e.g. DataID, position of CRC in data array) has to be identical on sender and receiver side.

The configuration of E2E State Machine is static as well and has to be provided by caller via function parameter.

7 Glossary and Abbreviations

7.1 Glossary

Term	Description
E2E	End to end protection

Table 7-1 Glossary

7.2 Abbreviations

Abbreviation	Description
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
BSW	Basis Software
CRC	Cyclic Redundancy Check
DEM	Diagnostic Event Manager
DET	Development Error Tracer
EAD	Embedded Architecture Designer
ECU	Electronic Control Unit
HIS	Hersteller Initiative Software
ISR	Interrupt Service Routine
MICROSAR	Microcontroller Open System Architecture (the Vector AUTOSAR solution)
PPORT	Provide Port
RPORT	Require Port
RTE	Runtime Environment
SRS	Software Requirement Specification
SWC	Software Component
SWS	Software Specification

Table 7-2 Abbreviations

8 Contact

Visit our website for more information on

> News
> Products
> Demo software
> Support
> Training data
> Addresses

www.vector.com