MICROSAR Classic COM Based Transformer

Technical Reference

Version 4.35.0

Authors	visrco, visso, viskba, viscfr, vislsa, vissi, jkugler
Status	Released

Document Information

Reference Documents

No.	Source	Title	Version
[1]	AUTOSAR	Specification of COM Based Transformer	R20-11
[2]	AUTOSAR	List of Basic Software Modules	R20-11
[3]	AUTOSAR	Specification of Communication	R20-11

Scope of the Document

This technical reference describes the general use of the COM Based Transformer.

!

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 Introduction

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

Supported Configuration Variants:	pre-compile	
Vendor ID:	COMXF_VENDOR_ID	30 decimal (= Vector-Informatik, according to HIS)
Module ID:	COMXF_MODULE_ID	175 decimal (according to ref. [2])

The ComXf module provides the functionality to serialize complex data when the target bus system uses a fixed communication matrix.

1.1 Architecture Overview

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

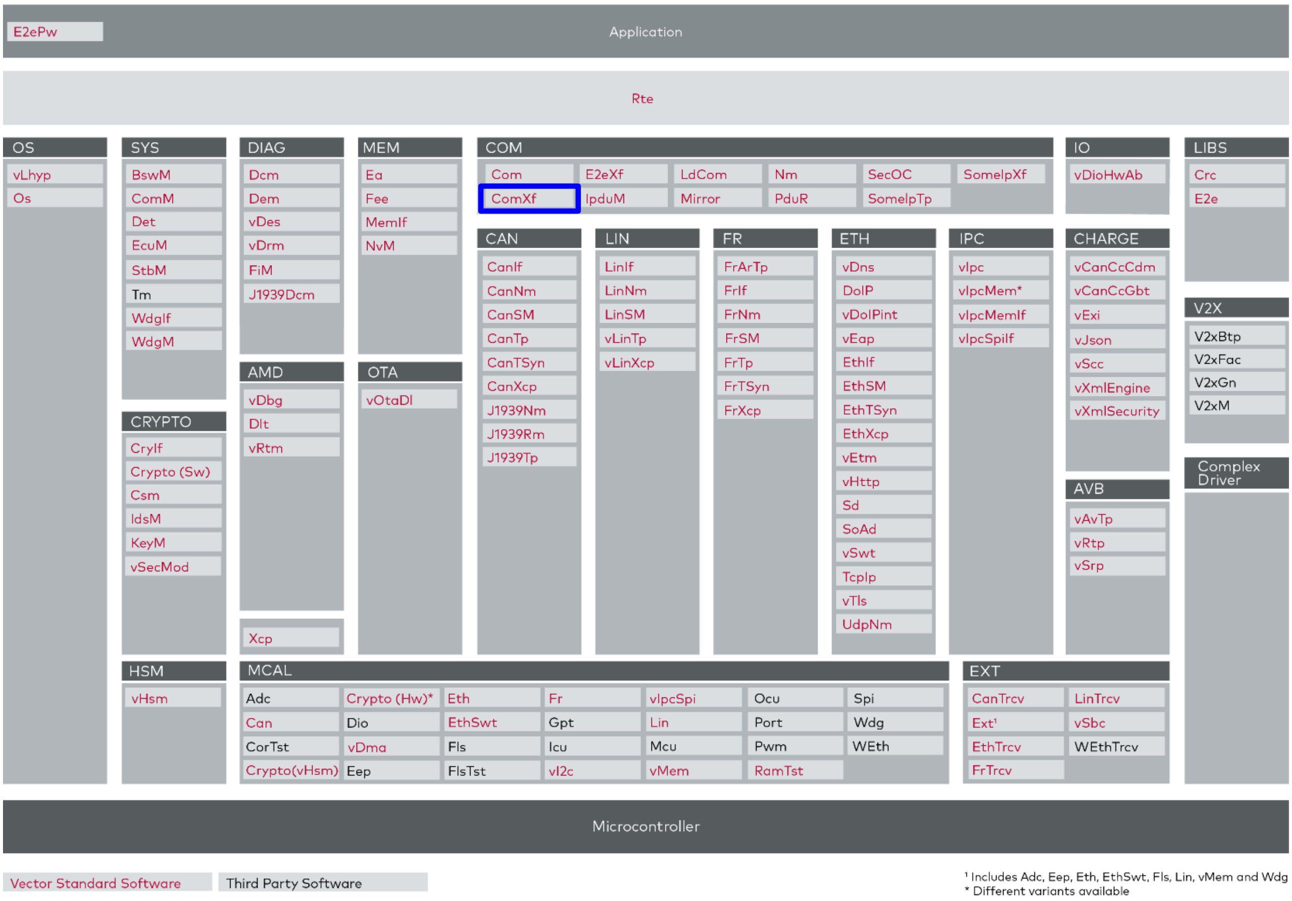


Figure 1-1 AUTOSAR Architecture Overview

2 Functional Description

2.1 Features

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

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

> Table 2-1 Supported AUTOSAR standard conform features

> Table 2-2 Not supported AUTOSAR standard conform features

The following features specified in [1] are supported:

Supported AUTOSAR Standard Conform Features
Serialization / Deserialization of complex data for S/R communication.
Handling of gaps in the array representation of a signal group.

Table 2-1 Supported AUTOSAR standard conform features

2.1.1 Deviations

The following features specified in [1] are not supported:

Category	Description
Functional	Development error detection.
Functional	Postbuild support.

Table 2-2 Not supported AUTOSAR standard conform features

2.2 Initialization

The ComXf does not have to be initialized or deinitialized. Calls to `ComXf_Init()` and `ComXf_DeInit()` can be omitted.

2.3 States

No internal states exist.

2.4 Main Functions

No main function exists because all functionality is performed within the called API.

2.5 Error Handling

2.5.1 Development Error Reporting

No development error reporting is currently supported by the ComXf.

2.5.2 Production Code Error Reporting

No production errors are specified for the ComXf.

3 Integration

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

3.1 Embedded Implementation

The delivery of the ComXf consists of:

File Name	Description	Integration Tasks
ComXf.c	Generated source file of the ComXf module.	-
ComXf.h	Generated main header file which shall be included by modules using the ComXf module.	-
ComXf_MemMap.h	Generated file with template areas that can be adapted by the user. It contains the ComXf specific part of the memory mapping.	Adapt the dedicated code areas within that file. See hints within that file.
ComXf_Compiler_Cfg.h	Generated file with template areas that can be adapted by the user. It contains the ComXf specific part of the compiler abstraction.	Adapt the dedicated code areas within that file. See hints within that file.

Table 3-1 Implementation files

4 API Description

4.1 Services provided by ComXf

4.1.1 ComXf_Init

Prototype	
`void` **`ComXf_Init`** `(const ComXf_ConfigType *config)`	
Parameter	
`config`	Pointer to the transformer's configuration data.
Return code	
`void`	none
Functional Description	
Initialization function.	
Particularities and Limitations	
none	
Expected Caller Context	
This function can be called in any context.	

Table 4-1 ComXf_Init

4.1.2 ComXf_DeInit

Prototype	
`void` **`ComXf_DeInit`** `(void)`	
Parameter	
`void`	none
Return code	
`void`	none
Functional Description	
Deinitialization function.	
Particularities and Limitations	
none	
Expected Caller Context	
This function can be called in any context.	

Table 4-2 ComXf_DeInit

4.1.3 ComXf_GetVersionInfo

Prototype	
void **ComXf_GetVersionInfo** (Std_VersionInfoType *versioninfo)	
Parameter	
versioninfo	Pointer to where to store the version information of this module.
Return code	
void	none
Functional Description	
This API returns version information, vendor ID and AUTOSAR module ID of the called transformer module.	
Particularities and Limitations	
This API is only available if enabled by the configuration parameter XfrmVersionInfoApi.	
Expected Caller Context	
This function can be called in any context.	

Table 4-3 ComXf_GetVersionInfo

4.1.4 ComXf_<transformerId>

Prototype	
`Std_ReturnType` **`ComXf_<transformerId>`** `(uint8 *buffer, uint32 *bufferLength, const <type> *dataElement)`	
Parameter	
`buffer`	Buffer allocated by the RTE, where the transformed data has to be stored by the transformer.
`bufferLength`	Used length of the buffer.
`dataElement`	Data element which shall be transformed.
Return code	
`E_OK`	Serialization successful.
Functional Description	
Serialization of signal group based on the COM ECU configuration for S/R communication.	
Particularities and Limitations	
none	
Expected Caller Context	
This function can be called in any context.	

Table 4-4 ComXf_<transformerId>

Caution
The return code `E_SER_GENERIC_ERROR` cannot occur and is therefore not supported.

4.1.5 ComXf_Inv_<transformerId>

Prototype	
`Std_ReturnType` **`ComXf_Inv_<transformerId>`** `(const uint8 *buffer, uint32 bufferLength, <type> *dataElement)`	
Parameter	
`buffer`	Buffer allocated by the RTE, where the serialized data is stored by the Rte.
`bufferLength`	Used length of the buffer.
`dataElement`	Data element which is the result of the transformation and contains the deserialized data element.
Return code	
`E_OK`	Deserialization successful.
Functional Description	
Deserialization of signal group based on the COM ECU configuration for S/R communication.	
Particularities and Limitations	
none	
Expected Caller Context	
This function can be called in any context.	

Table 4-5 ComXf_Inv_<transformerId>

Caution
The return code `E_SER_GENERIC_ERROR` cannot occur and is therefore not supported.

Caution
The MICROSAR Classic RTE never calls `ComXf_Inv_<id>` with a `NULL_PTR` or with a buffer size equal 0. Therefore, the return code `E_NO_DATA` cannot occur and is thus not supported.

5 Configuration

In the ComXf the attributes can be configured with the following tools:

> Configuration in DaVinci Configurator

Currently, only the GetVersionInfo API can be enabled / disabled in the ComXf ECU configuration.
The serialization / deserialization of the signal groups is based on the ECU configuration of the COM module (for details see [3]).

5.1 Configuration Variants

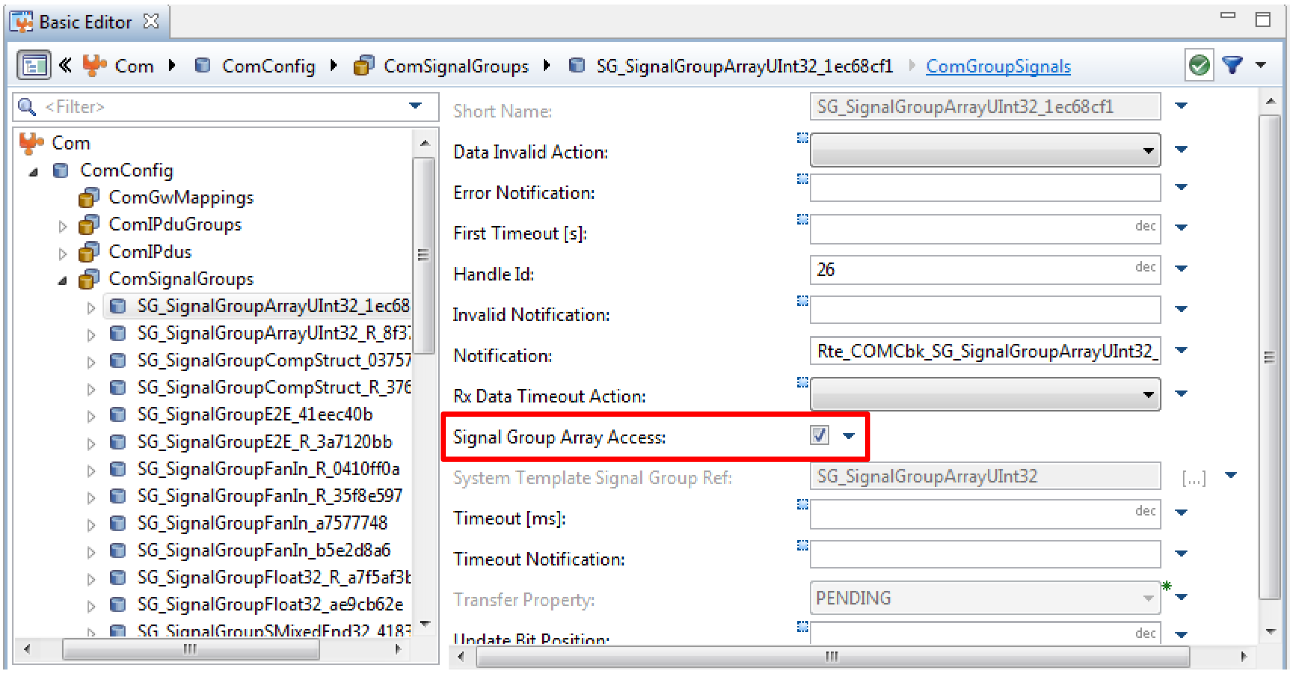
The ComXf supports the configuration variants

> `VARIANT-PRE-COMPILE`

The configuration classes of the ComXf parameters depend on the supported configuration variants. For their definitions please see the `ComXf_bswmd.arxml` file.

5.2 Enabling / Disabling of data transformation

If a signal shall be handled by the ComXf, the parameter “Signal Group Array Access” has to be set in the ECU configuration of the COM module (see Figure 5-1 Enable Data Transformation).


Basic Editor
Com
ComConfig
ComSignalGroups
SG_SignalGroupArrayUInt32_1ec68cf1
ComGroupSignals
<Filter>
Com
ComConfig
ComGwMappings
ComIPduGroups
ComIPdus
ComSignalGroups
SG_SignalGroupArrayUInt32_1ec68
SG_SignalGroupArrayUInt32_R_8f3
SG_SignalGroupCompStruct_03757
SG_SignalGroupCompStruct_R_376
SG_SignalGroupE2E_41eec40b
SG_SignalGroupE2E_R_3a7120bb
SG_SignalGroupFanIn_R_0410ff0a
SG_SignalGroupFanIn_R_35f8e597
SG_SignalGroupFanIn_a7577748
SG_SignalGroupFanIn_b5e2d8a6
SG_SignalGroupFloat32_R_a7f5af3b
SG_SignalGroupFloat32_ae9cb62e
SG_SignalGroupSMixedEnd32_4183
dec
[...]

Short Name: SG_SignalGroupArrayUInt32_1ec68cf1

Data Invalid Action:

Error Notification:

First Timeout [s]:

Handle Id: 26

Invalid Notification:

Notification: Rte_COMCbk_SG_SignalGroupArrayUInt32_

Rx Data Timeout Action:

Signal Group Array Access: ☑

System Template Signal Group Ref: SG_SignalGroupArrayUInt32

Timeout [ms]:

Timeout Notification:

Transfer Property: PENDING

Update Bit Position:

Figure 5-1 Enable Data Transformation

6 Glossary and Abbreviations

6.1 Glossary

Term	Description
DaVinci Configurator	Configuration and generation tool for MICROSAR Classic components

Table 6-1 Glossary

6.2 Abbreviations

Abbreviation	Description
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
BSW	Basis Software
DEM	Diagnostic Event Manager
DET	Development Error Tracer
ECU	Electronic Control Unit
MICROSAR	Microcontroller Open System Architecture (the Vector AUTOSAR solution)
RTE	Runtime Environment
SRS	Software Requirement Specification
SWC	Software Component
SWS	Software Specification

Table 6-2 Abbreviations

7 Additional Copyrights

The MICROSAR Classic COMXF Generator contains *Free and Open Source Software* (FOSS). The following table lists the files which contain this software, the kind and version of the FOSS, the license under which this FOSS is distributed and a reference to a license file which contains the original text of the license terms and conditions. The referenced license files can be found in the directory of the RTE Generator.

File	FOSS	License	License Reference
MicrosarComXfGen64.exe	Perl 5.30	Artistic License	License_Artistic.txt

Table 7-1 Free and Open Source Software Licenses

8 Contact

Visit our website for more information on

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

www.vector.com