MICROSAR Vsg

Technical Reference

Version 2.00.00

Authors	Savas Ates
Status	Released

Document Information

History

Author	Date	Version	Remarks
S. Ates	2014-05-08	1.0.0	Document creation
S. Ates	2015-09-22	2.0.0	Rework of initialization concept

Reference Documents

No.	Source	Title	Version
[1]	AUTOSAR	AUTOSAR_SWS_DET.pdf	V4.2.0
[2]	AUTOSAR	AUTOSAR_SWS_DEM.pdf	V4.2.0
[3]	AUTOSAR	AUTOSAR_BasicSoftwareModules.pdf	V1.0.0
[4]	Vector	TechnicalReference_Diag_Asr4Dem_Generic.pdf	V3.2.0
[5]	Vector	TechnicalReference_CANdesc	V3.7.0

Scope of the Document**:**

This technical reference describes the general use of the Cdd_Vsg software.

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 Version
2.00.00	Rework Initialization Concept

Table 1-1 Component history

2 Introduction

This document describes the functionality, API and configuration of the MICROSAR BSW module Cdd_Vsg.

Supported AUTOSAR Release*:	4	
Supported Configuration Variants:	pre-compile	
Vendor ID:	VSG_VENDOR_ID	30 decimal (= Vector-Informatik, according to HIS)
Module ID:	VSG_MODULE_ID	255 decimal (according to ref. [3])

* For the precise AUTOSAR Release 4.x please see the release specific documentation.

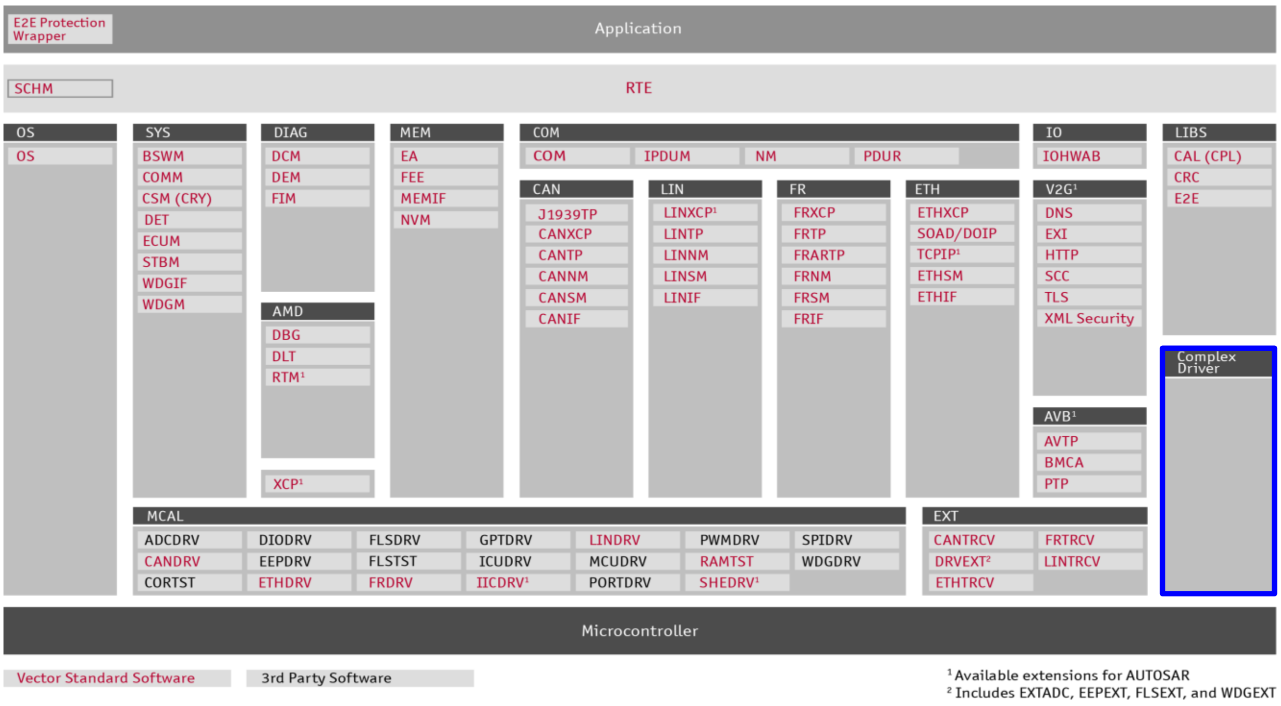
The Vsg is a module to support different diagnostic configurations in a car. VSGs are also called “dependencies”.

2.1 How to read this document

In this documentation the MICROSAR Vsg module will be called Cdd_Vsg. Thus should make it easy to distinguish linguistically between the MICROSAR Vsg module and VSG as Vehicle System Group.

2.2 Architecture Overview

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



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

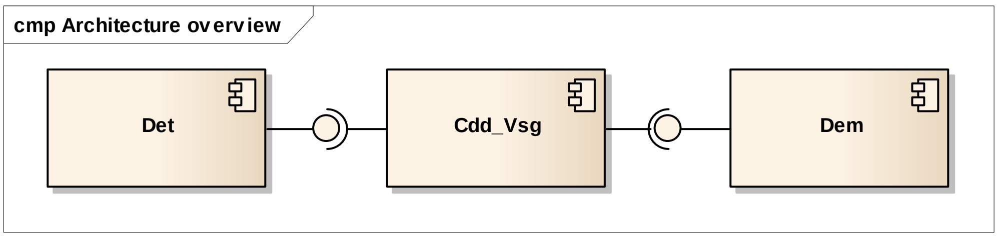


Figure 2-1 Interfaces to adjacent modules of the Cdd_Vsg

3 Functional Description

3.1 VSG

A VSG provides the possibility to group a set of diagnostic events. The availability of these diagnostic events can be changed by enabling or disabling the associated VSG.

The APIs `Vsg_EnableVsg()` and `Vsg_DisableVsg()` allow to enable/disable individual required VSGs and the associated diagnostic events during runtime after the initialization of the Cdd_Vsg.

Enabling a VSG additionally sets the internal status of a VSG to *active* if all associated diagnostic events can be enabled successfully.

Disabling a VSG sets the internal status of a VSG always to *inactive* even if one or more associated diagnostic events cannot be disabled.

The API `Vsg_Finalize()` allows to disable all *inactive* VSGs.

During initialization only the internal status of each VSG is marked as *inactive* without disabling the VSGs and the associated diagnostic events.

During Shutdown all *inactive* VSGs will be disabled.

3.2 Features

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

The following feature is supported:

Supported Features
Enable VSG
Enabling a single VSG using the service `Vsg_EnableVsg()`.
Disable VSG
Disable a single VSG using the service `Vsg_DisableVsg()`.
Finalize VSGs
Disables all inactive VSGs using the service `Vsg_Finalize()`.

Table 3-1 Supported feature

3.3 Initialization

The interface `Vsg_Init()` sets the internal Status of each VSG to inactive.

3.4 Error Handling

3.4.1 Development Error Reporting

By default, development errors are reported to the DET using the service `Det_ReportError()` as specified in [1], if development error reporting is enabled (i.e. pre-compile parameter `VSG_DEV_ERROR_DETECT==STD_ON`).

The reported module ID for the module Cdd_Vsg is 255 (Complex Device Driver).

The reported service IDs identify the services which are described in 5.2. The following table presents the service IDs and the related services:

Service ID	Service
0x00	Vsg_GetVersionInfo
0x02	Vsg_EnableVsg
0x03	Vsg_DisableVsg
0x04	Vsg_Finalize
0x20	Vsg_Shutdown

Table 3-2 Service IDs

The errors reported to DET are described in the following table:

Error Code	Description
VSG_E_PARAM_POINTER	Service called with an invalid NULL pointer argument.
VSG_E_PARAM_DATA	Service was called with invalid parameter value.
VSG_E_UNINIT	Service called in uninitialized state.

Table 3-3 Errors reported to DET

4 Integration

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

VSGs in the diagnostic kernel have to be handled separately. For reference see [5] .

4.1 Shutdown

The interface `Vsg_Shutdown()` has to be called before the shutdown of the Dem module (see [4]) . Otherwise configured diagnostic events that are associated to a VSG will not be disabled at shutdown.

4.2 Scope of Delivery

The delivery of the Cdd_Vsg contains the files which are described in the chapters 4.2.1 and 4.2.2:

4.2.1 Static Files

The delivery of the Cdd_Vsg does not contain static files.

4.2.2 Dynamic Files

The dynamic files are generated by the configuration tool DaVinci Configurator Pro.

File Name	Description
Vsg.h	This header files provides the Cdd_Vsg API functions. For BSW modules and the application. This file is supposed to be included by client modules.
Vsg.c	This is the source file of the Cdd_Vsg. It contains all functionality of the Cdd_Vsg

Table 4-1 Generated files

4.3 Include Structure

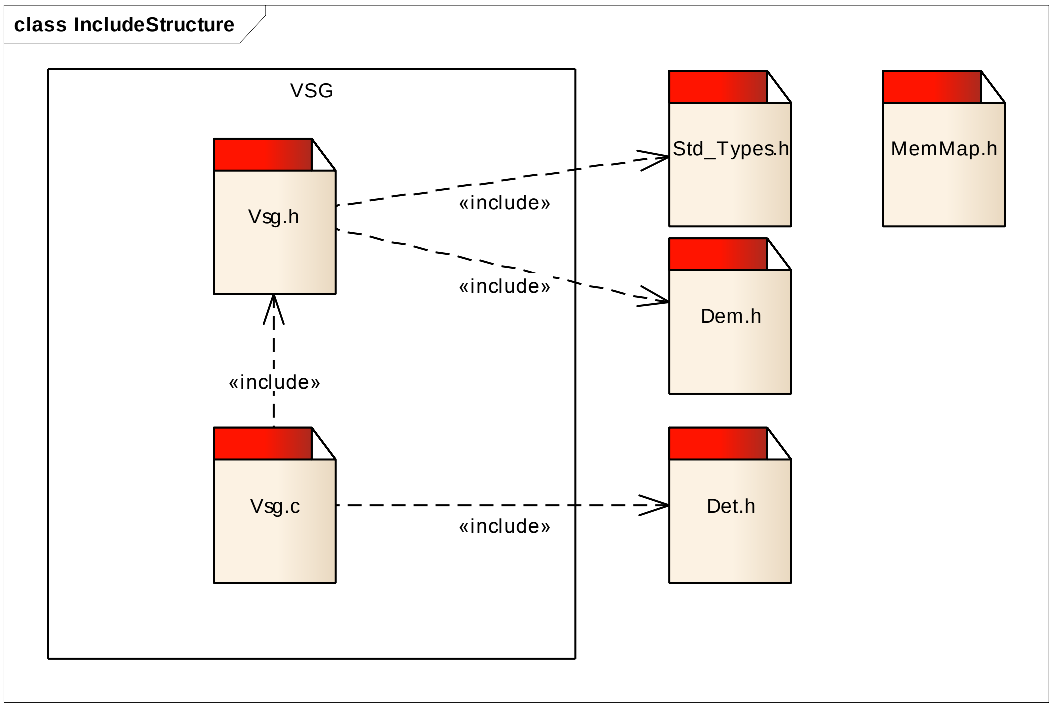


Figure 4-1 Include structure

4.4 Compiler Abstraction and Memory Mapping

The objects (e.g. variables, functions, constants) are declared by compiler independent definitions – the compiler abstraction definitions. Each compiler abstraction definition is assigned to a memory section.

The following table contains the memory section names and the compiler abstraction definitions of the Cdd_Vsg and illustrates their assignment among each other.

Compiler Abstraction Definitions / Memory Mapping Sections	VSG_CODE	VSG_CONST	VSG_VAR_INIT	VSG_VAR_NOINIT	VSG_APPL_DATA
VSG_START_SEC_CODE VSG_STOP_SEC_CODE	■				
VSG_START_SEC_CONST_<size> VSG_STOP_SEC_CONST_<size>		■			
VSG_START_SEC_VAR_INIT_<size> VSG_STOP_SEC_VAR_INIT_<size>			■		

VSG_START_SEC_VAR_NOINIT_UNSPECIFIED VSG_STOP_SEC_VAR_NOINIT_UNSPECIFIED				■	
Application buffer used in API					■

Table 4-2 Compiler abstraction and memory mapping

4.5 Critical Sections

There are no critical sections defined for the Cdd_Vsg.

5 API Description

For an interface overview please see Figure 2-1.

5.1 Type Definitions

The types defined by the Cdd_Vsg are described in this chapter.

Type Name	C-Type	Description	Value Range
Vsg_VsgItemIdType	uint8	Unique identification of a VSG	`0..31`

Table 5-1 Type definitions

5.2 Services provided by Cdd_Vsg

5.2.1 Vsg_EnableVsg()

Prototype	
`Std_ReturnType` **`Vsg_EnableVsg`**`(Vsg_VsgItemIdType VsgItemId)`	
Parameter	
`VsgItemId`	Unique identification of a VSG (Vehicle System Group).
Return code	
`Std_ReturnType`	E_OK: operation was successful E_NOT_OK: availability of one or more events could not be enabled
Functional Description	
API to enable a single Vehicle System Group.	
Particularities and Limitations	
> This function can be called from any context. > This function is reentrant (for different VsgItemId with disjoint set of diagnostic events). > This function is not reentrant with other Services provided by Cdd_Vsg. > This function is synchronous.	

Table 5-2 Vsg_EnableVsg()

5.2.2 Vsg_DisableVsg()

Prototype	
`Std_ReturnType` **`Vsg_DisableVsg`**`(Vsg_VsgItemIdType VsgItemId)`	
Parameter	
`VsgItemId`	Unique identification of a VSG (Vehicle System Group).

Return code	
`Std_ReturnType`	E_OK: operation was successful E_NOT_OK: availability of one or more events could not be disabled
Functional Description	
API to disable a single Vehicle System Group.	
Particularities and Limitations	
> This function can be called from any context. > This function is reentrant (for different VsgItemId with disjoint set of diagnostic events). > This function is not reentrant with other Services provided by Cdd_Vsg. > This function is synchronous.	

Table 5-3 Vsg_DisableVsg()

5.2.3 Vsg_Finalize()

Prototype	
`Std_ReturnType` **`Vsg_Finalize`** `(void)`	
Parameter	
`N/A`	N/A
Return code	
`Std_ReturnType`	E_OK: operation was successful E_NOT_OK: one or more VSGs could not be disabled successful
Functional Description	
All VSGs with internal status inactive are disabled.	
Particularities and Limitations	
> This function can be called from any context. > This function is not reentrant. > This function is synchronous.	

Table 5-4 Vsg_Finalize()

5.2.4 Vsg_Init()

Prototype	
`void` **`Vsg_Init`** `(void)`	
Parameter	
`N/A`	N/A
Return code	
`void`	N/A

Functional Description
Internal status of all VSGs is set to inactive.
Particularities and Limitations
> This function can be called from any context. > This function is not reentrant. > This function is synchronous.

Table 5-5 Vsg_Init()

5.2.5 Vsg_Shutdown()

Prototype	
`void` **`Vsg_Shutdown`**`(void)`	
Parameter	
`N/A`	N/A
Return code	
`void`	N/A
Functional Description	
Shutdown Cdd_Vsg functionality. All VSGs with internal status inactive are disabled.	
Particularities and Limitations	
> This function can be called from any context. > This function is not reentrant. > This function is synchronous.	

5.3 Services used by Cdd_Vsg

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

Component	API
Dem	Dem_SetEventAvailable

Table 5-6 Services used by the Cdd_Vsg

5.4 Callback Functions

There are no callback functions implemented by the Cdd_Vsg.

6 Configuration

The Cdd_Vsg module is configured with the help of the configuration tool DaVinci Configurator Pro.

6.1 Configuration Variants

The Cdd_Vsg supports the configuration variants

> `VARIANT-PRE-COMPILE`

The configuration classes of the Cdd_Vsg parameters depend on the supported configuration variants. For their definitions please see the Vsg_bswmd.arxml file.

6.2 Configuration Attributes

The description of each configurable option is described within the Vsg_bswmd.arxml file. You can use the online help of DaVinci Configurator Pro to access these parameter descriptions comfortably.

7 Glossary and Abbreviations

7.1 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)
SWC	Software Component
VSG	Vehicle System Group

Table 7-1 Abbreviations

8 Contact

Visit our website for more information on

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

www.vector.com