MICROSAR Vsg

Technical Reference

Version 3.00.00

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
S. Ates	2017-03-12	3.0.0	VSG support Dcm

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_Dem.pdf	V7.0.0
[5]	Vector	TechnicalReference_Dcm.pdf	V7.2.0
[6]	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

1 Component History 6

2 Introduction 7
2.1 How to read this document 7
2.2 Architecture Overview 7

3 Functional Description 9
3.1 VSG 9
3.2 Features 9
3.3 Initialization 10
3.4 Error Handling 10
3.4.1 Development Error Reporting 10

4 Integration 11
4.1 Shutdown 11
4.2 Scope of Delivery 11
4.2.1 Static Files 11
4.2.2 Dynamic Files 11
4.3 Include Structure 12
4.4 Compiler Abstraction and Memory Mapping 12
4.5 Critical Sections 13

5 API Description 14
5.1 Type Definitions 14
5.2 Services provided by Cdd_Vsg 14
5.2.1 Vsg_EnableVsg() 14
5.2.2 Vsg_DisableVsg() 15
5.2.3 Vsg_EnableVsgMultiple() 16
5.2.4 Vsg_DisableVsgMultiple() 16
5.2.5 Vsg_IsVsgActive() 17
5.2.6 Vsg_IsAnyVsgActive() 18
5.2.7 Vsg_Finalize() 18
5.2.8 Vsg_GetVersionInfo() 19
5.2.9 Vsg_Init() 19
5.2.10 Vsg_InitMemory() 19
5.2.11 Vsg_Shutdown() 20
5.3 Services used by Cdd_Vsg 20
5.4 Callback Functions 21

6 Configuration 22
6.1 Configuration Variants 22
6.2 Configuration Attributes 22

7 Glossary and Abbreviations 23
7.1 Abbreviations 23

8 Contact 24

Illustrations

Figure 2-1 AUTOSAR Architecture Overview 7
Figure 2-2 Interfaces to adjacent modules of the Cdd_Vsg 8
Figure 4-1 Include structure 12

Tables

Table 1-1 Component history 6
Table 3-1 Supported features 9
Table 3-2 Service IDs 10
Table 3-3 Errors reported to DET 10
Table 4-1 Generated files 11
Table 4-2 Compiler abstraction and memory mapping 13
Table 5-1 Type definitions 14
Table 5-2 Vsg_EnableVsg() 15
Table 5-3 Vsg_DisableVsg() 15
Table 5-4 Vsg_EnableVsgMultiple() 16
Table 5-5 Vsg_DisableVsgMultiple() 17
Table 5-6 Vsg_IsVsgActive() 17
Table 5-7 Vsg_IsAnyVsgActive() 18
Table 5-8 Vsg_Finalize() 18
Table 5-9 Vsg_GetVersionInfo() 19
Table 5-10 Vsg_Init() 19
Table 5-11 Vsg_InitMemory() 20
Table 5-12 Vsg_Shutdown 20
Table 5-13 Services used by the Cdd_Vsg 20
Table 7-1 Abbreviations 23

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
3.00.00	Breaking change in Data model of DaVinci Configurator
4.00.00	VSG support Dcm

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.

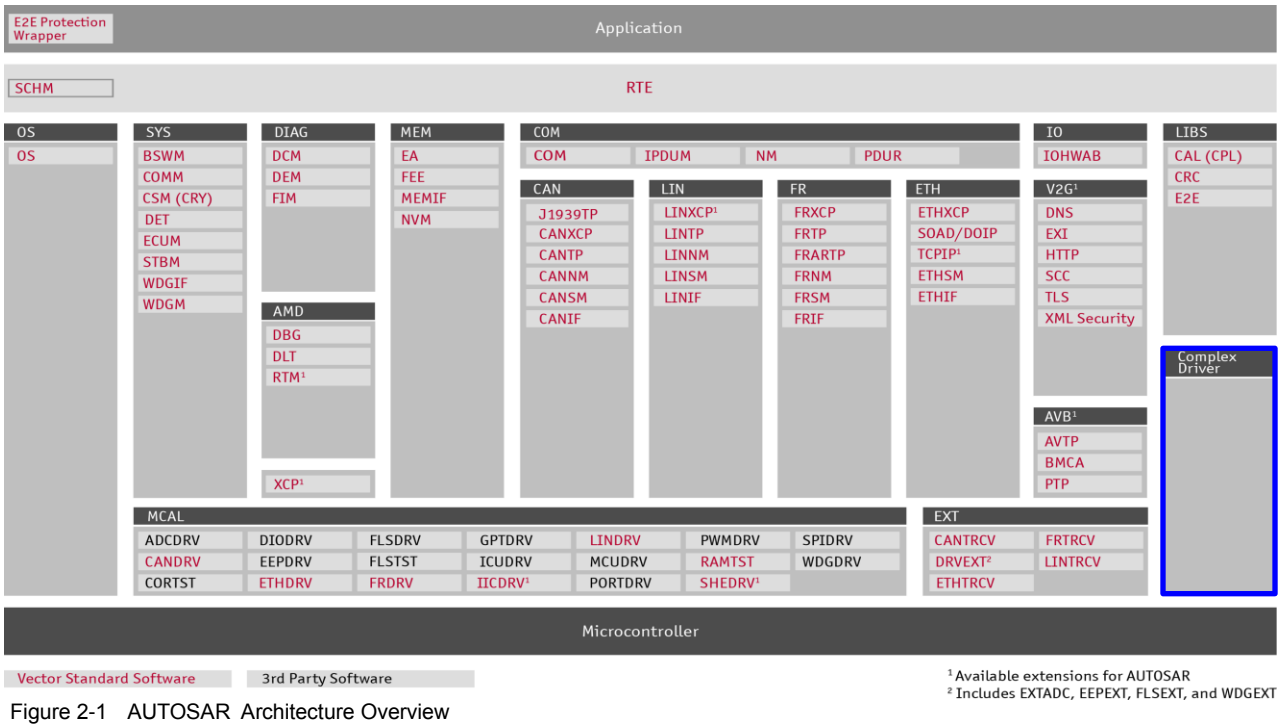

Figure 2-1 AUTOSAR Architecture Overview

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

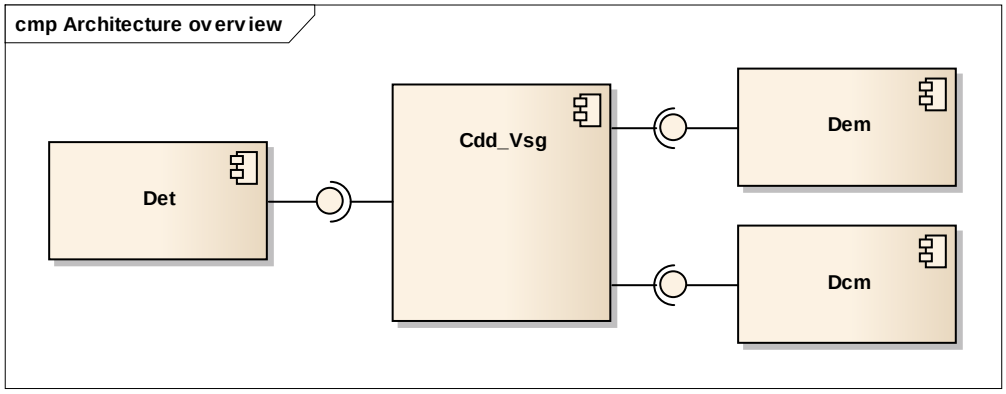

Figure 2-2 Interfaces to adjacent modules of the Cdd_Vsg

3 Functional Description

3.1 VSG

A VSG provides the possibility to group a set of diagnostic objects. The availability of these diagnostic objects 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 objects during runtime after the initialization of the Cdd_Vsg.

Enabling a VSG sets the status of a VSG to *active* if all associated diagnostic objects are enabled successfully. If enabling of at least one diagnostic object fails the status of VSG is set to *undefined.* Status of VSG is not changed if no diagnostic object is enabled.

Disabling a VSG sets the status of a VSG to in*active* if all associated diagnostic objects are enabled successfully. If disabling of at least one diagnostic object fails the status of VSG is set to *undefined.* Status of VSG is not changed if no diagnostic object is disabled.

During initialization the status of each VSG is set to *undefined*. Therefore after initialization it is recommended to enable required VSGs and use the API `Vsg_Finalize()` to disable all VSGs with *undefined* status along with their associated diagnostic objects.

Undefined VSGs will be disabled during shutdown of the Cdd_Vsg.

3.2 Features

The features listed in the following table cover the complete functionality specified for the Cdd_Vsg:

Supported Features
Enable VSG
Enabling a single VSG using the service `Vsg_EnableVsg()`.
Enabling multiple VSGs using the service `Vsg_EnableVsgMultiple()`.
Disable VSG
Disabling a single VSG using the service `Vsg_DisableVsg()`.
Disabling multiple VSGs using the service `Vsg_DisableVsgMultiple()`.
Query VSG status
Query VSG status using the service `Vsg_IsVsgActive()` or `Vsg_IsAnyVsgActive()`.
Finalize VSGs
Disabling all VSGs with status undefined using the service `Vsg_Finalize()`.

Table 3-1 Supported features

3.3 Initialization

The interface `Vsg_Init()` sets the status of each VSG to undefined.

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
0x05	Vsg_IsVsgActive
0x06	Vsg_IsAnyVsgActive
0x07	Vsg_EnableVsgMultiple
0x08	Vsg_DisableVsgMultiple
0x20	Vsg_SetVsgMultiple (Internal Function)
0x30	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 [6].

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 file 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

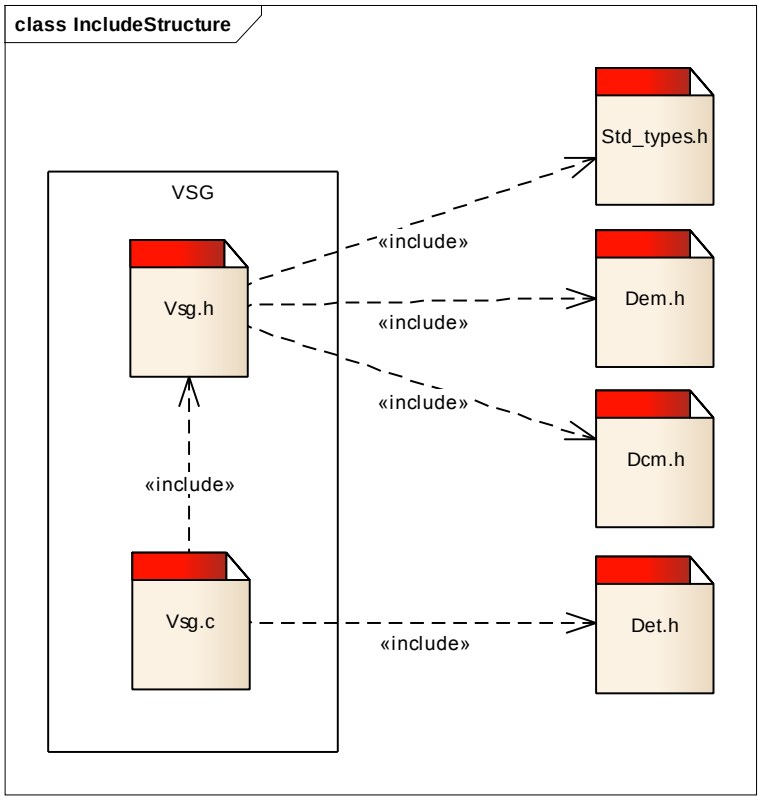

Figure 4-1 Include structure

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 2-2.

5.1 Type Definitions

The types defined by the Cdd_Vsg are described in Table 5-1.

Type Name	C-Type	Description	Value Range
Vsg_VsgItemIdType	uint16	Unique identification of a VSG	`0..65535`
Vsg_VsgItemIdSizeType	uint16	Number of VSGs	`0..65535`
Vsg_VsgStateType	uint8	VSG status type	`VSG_VSGENABLED` VSG status is active
			`VSG_VSGDISABLED` VSG status is inactive

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, VSG status is now active. E_NOT_OK: one or more diagnostic objects could not be enabled. VSG status is undefined if at least one diagnostic object is enabled, otherwise VSG status is not modified.
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 VsgItemIds). > 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, VSG status is now inactive. E_NOT_OK: one or more diagnostic objects could not be disabled. VSG status is undefined if at least one diagnostic object is disabled, otherwise VSG status is not modified.
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 VsgItemIds). > 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_EnableVsgMultiple()

Prototype	
`Std_ReturnType` **`Vsg_EnableVsgMultiple`**`(Vsg_VsgItemIdType* VsgItemList, Vsg_VsgItemIdSizeType NumOfVsgItems)`	
Parameter	
`VsgItemList`	Pointer to a list of VSGs (Vehicle System Groups).
`NumOfVsgItems`	Number of VSGs in `VsgItemList`.
Return code	
`Std_ReturnType`	E_OK: operation was successful. E_NOT_OK: one or more VSGs could not be enabled.
Functional Description	
API to enable multiple Vehicle System Groups.	
Particularities and Limitations	
> This function can be called from any context. > This function is reentrant (for different VsgItemIds). > This function is not reentrant with other Services provided by Cdd_Vsg. > This function is synchronous.	

Table 5-4 Vsg_EnableVsgMultiple()

Caution
Vehicle System Groups that reference a DCM object shall only exist once in the passed `VsgItemList`. Otherwise a buffer overflow can occur. This error will be reported if DET is used (see 3.4 Error Handling).

5.2.4 Vsg_DisableVsgMultiple()

Prototype	
`Std_ReturnType` **`Vsg_DisableVsgMultiple`**`(Vsg_VsgItemIdType* VsgItemList, Vsg_VsgItemIdSizeType NumOfVsgItems)`	
Parameter	
`VsgItemList`	Pointer to a list of VSGs (Vehicle System Groups).
`NumOfVsgItems`	Number of VSGs in `VsgItemList`.

Return code	
`Std_ReturnType`	E_OK: operation was successful. E_NOT_OK: one or more VSGs could not be disabled.
Functional Description	
API to disable multiple Vehicle System Groups.	
Particularities and Limitations	
> This function can be called from any context. > This function is reentrant (for different VsgItemIds). > This function is not reentrant with other Services provided by Cdd_Vsg. > This function is synchronous.	

Table 5-5 Vsg_DisableVsgMultiple()

Caution
Vehicle System Groups that reference a DCM object shall only exist once in the passed `VsgItemList`. Otherwise a buffer overflow can occur. This error will be reported if DET is used (see 3.4 Error Handling).

5.2.5 Vsg_IsVsgActive()

Prototype	
`Std_ReturnType` **`Vsg_IsVsgActive`**`(Vsg_VsgItemIdType VsgItemId, Vsg_VsgStateType* IsActive)`	
Parameter	
`VsgItemId`	Unique identification of a VSG (Vehicle System Group).
`IsActive`	Provides the status of VSG if return code is E_OK.
Return code	
`Std_ReturnType`	E_OK: operation was successful. E_NOT_OK: operation failed.
Functional Description	
API to query current status of a single VSG.	
Particularities and Limitations	
> This function can be called from any context. > This function is reentrant. > This function is synchronous.	

Table 5-6 Vsg_IsVsgActive()

5.2.6 Vsg_IsAnyVsgActive()

Prototype	
`Std_ReturnType` **`Vsg_IsAnyVsgActive`**`(Vsg_VsgItemIdType* VsgItemList, Vsg_VsgItemIdSizeType NumOfVsgItems, Vsg_VsgStateType* IsActive)`	
Parameter	
`VsgItemList`	Pointer to a list of VSGs (Vehicle System Groups).
`NumOfVsgItems`	Number of VSGs in `VsgItemList`.
`IsActive`	If return code is E_OK parameter IsActive provides the status VSG_VSGENABLED if at least one VSG is active and VSG_VSGDISABLED otherwise.
Return code	
`Std_ReturnType`	E_OK: operation was successful. E_NOT_OK: operation failed.
Functional Description	
API to query if status of at least one VSG in a list of VSGs is active.	
Particularities and Limitations	
> This function can be called from any context. > This function is reentrant. > This function is synchronous.	

Table 5-7 Vsg_IsAnyVsgActive()

5.2.7 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 successfully.
Functional Description	
All VSGs with undefined status are disabled.	
Particularities and Limitations	
> This function can be called from any context. > This function is not reentrant. > This function is synchronous.	

Table 5-8 Vsg_Finalize()

5.2.8 Vsg_GetVersionInfo()

Prototype	
`void Vsg_GetVersionInfo (Std_VersionInfoType* versioninfo)`	
Parameter	
`versioninfo`	Pointer to where to store the version information of this module.
Return code	
`void`	N/A
Functional Description	
Returns the version information of this module.	
Particularities and Limitations	
> This function can be called from any context. > This function is reentrant. > This function is synchronous.	

Table 5-9 Vsg_GetVersionInfo()

5.2.9 Vsg_Init()

Prototype	
`void Vsg_Init(void)`	
Parameter	
`N/A`	N/A
Return code	
`void`	N/A
Functional Description	
Status of all VSGs is set to undefined.	
Particularities and Limitations	
> This function can be called from any context. > This function is not reentrant. > This function is synchronous.	

Table 5-10 Vsg_Init()

5.2.10 Vsg_InitMemory()

Prototype	
`void Vsg_InitMemory(void)`	
Parameter	
`N/A`	N/A

Return code	
`void`	N/A
Functional Description	
Use this function to initialize RAM variables in case the start-up code is not used to initialize RAM.	
Particularities and Limitations	
> This function may not interrupt any other APIs. > This function is not reentrant. > This function is synchronous.	

Table 5-11 Vsg_InitMemory()

5.2.11 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 undefined status are disabled.	
Particularities and Limitations	
> This function can be called from any context. > This function is not reentrant. > This function is synchronous.	

Table 5-12 Vsg_Shutdown

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
Dcm	Dcm_VsgSetSingle
	Dcm_VsgSetMultiple
	Dcm_VsgIsActive

Table 5-13 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