MICROSAR FiM

Technical Reference

Version 2.1.4

Authors	Joachim Kalmbach, Katrin Thurow
Status	Released

1 Document Information

1.1 History

Author	Date	Version	Remarks
Joachim Kalmbach	2007-01-03	1.00.00	Initial version
Joachim Kalmbach	2007-08-21	1.01.00	Updated for AUTOSAR 2.1
Katrin Thurow	2007-10-10	1.02.00	Updated supported features and known issues
Katrin Thurow	2008-06-05	2.00.00	Updated for AUTOSAR 3
Katrin Thurow	2008-07-02	2.01.00	Change of error table
Katrin Thurow	2009-07-22	2.01.01	Modified not supported feature table
Katrin Thurow	2010-01-17	2.01.02	Small corrections Modified not supported feature table
Katrin Thurow	2010-11-30	2.01.03	Added chapter 4.4 Critical Sections
Katrin Thurow	2011-12-08	2.01.04	Changed chapter 4.4 Critical Sections

Table 1-1 History of the document

1.2 Reference Documents

No.	Title	Version
[1]	AUTOSAR_SWS_FIM.pdf	V1.2.0
[2]	AUTOSAR_BasicSoftwareModules.pdf	V1.0.0

Table 1-2 Reference documents

Please note
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 Document Information 2
1.1 History 2
1.2 Reference Documents 2

Component History 7

2 Introduction 8
2.1 Architecture Overview 9

3 Functional Description 11
3.1 Features 11
3.2 Initialization 11
3.3 States 12
3.4 Main Functions 12
3.5 Error Handling 12
3.5.1 Development Error Reporting 12
3.5.2 Production Code Error Reporting 13

4 Integration 14
4.1 Scope of Delivery 14
4.1.1 Static Files 14
4.1.2 Dynamic Files 14
4.2 Include Structure 15
4.3 Compiler Abstraction and Memory Mapping 15
4.4 Critical Sections 16

5 API Description 17
5.1 Type Definitions 17
5.2 Services provided by FiM 18
5.2.1 Fim_Init 18
5.2.2 Fim_DemInit 19
5.2.3 Fim_GetFunctionPermission 19
5.2.4 Fim_DemTriggerOnEventStatus 20
5.2.5 Fim_GetVersionInfo 21
5.3 Services used by the Function Inhibition Manager 21
5.4 Service Ports 21
5.4.1 Client Server Interface 21
5.4.1.1 Provide Ports on FiM Side 21

5.4.1.1.1 FunctionInhibition 22

6 Configuration 23

6.1 Configuration with GENy 23

6.1.1 Pre-compile Properties 23

6.1.2 FunctionID specific Properties 26

7 AUTOSAR Standard Compliance 27

7.1 Deviations 27

7.2 Additions/ Extensions 27

7.3 Limitations 27

8 Glossary and Abbreviations 28

8.1 Glossary 28

8.2 Abbreviations 28

9 Contact 29

Illustrations

Figure 2-1 AUTOSAR layer model 9
Figure 2-2 Interfaces to adjacent modules of the FiM 10
Figure 3-1 Initialization sequence of the FiM 12
Figure 4-1 Include structure 15
Figure 6-1 GENy pre-compile configuration 23
Figure 6-2 GENy FID specific configuration 26

Tables

Table 1-1 History of the document 2
Table 1-2 Reference documents 2
Table 0-1 Component history 7
Table 3-1 Supported SWS features 11
Table 3-2 Not supported SWS features 11
Table 3-3 Mapping of service IDs to services 13
Table 3-4 Errors reported to DET 13
Table 4-1 Static files- 14
Table 4-2 Generated files 14
Table 4-3 Compiler abstraction and memory mapping 15
Table 5-1 Type definitions 17
Table 5-2 Fim_EventIdTableType 17
Table 5-3 Fim_EventIdTableType 18
Table 5-4 Fim_Init 18
Table 5-5 Fim_DemInit 19
Table 5-6 Fim_GetFunctionPermission 19
Table 5-7 Fim_GetFunctionPermission 20
Table 5-8 Fim_GetVersionInfo 21
Table 5-9 Services used by the FiM 21
Table 6-1 GENy pre-compile configuration 25
Table 6-2 GENy FID specific configuration 26
Table 8-1 Glossary 28
Table 8-2 Abbreviations 28

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
2.00	Update of the component to AUTOSAR 3: Compiler abstraction is introduced and additional checks of the version of the generated files.

Table 0-1 Component history

2 Introduction

This document describes the functionality, API and configuration of the AUTOSAR Function Inhibition Manager (FIM) as specified in [1].

Supported AUTOSAR Release*:	3	
Supported Configuration Variants:	pre-compile, link-time	
Vendor ID:	FIM_VENDOR_ID	30 decimal (= Vector-Informatik, according to HIS)
Module ID:	FIM_MODULE_ID	11 decimal (according to ref. [2])

* For the precise AUTOSAR Release 3.x please see the release specific documentation.

The Function Inhibition Manager is responsible for providing a control mechanism for software components and the functionality therein. In this context, functionality can be built up of the contents of one, several or parts of runnable entities with the same set of permission / inhibit conditions.

2.1 Architecture Overview

The following figure shows where the Function Inhibition Manager is located in the AUTOSAR architecture.

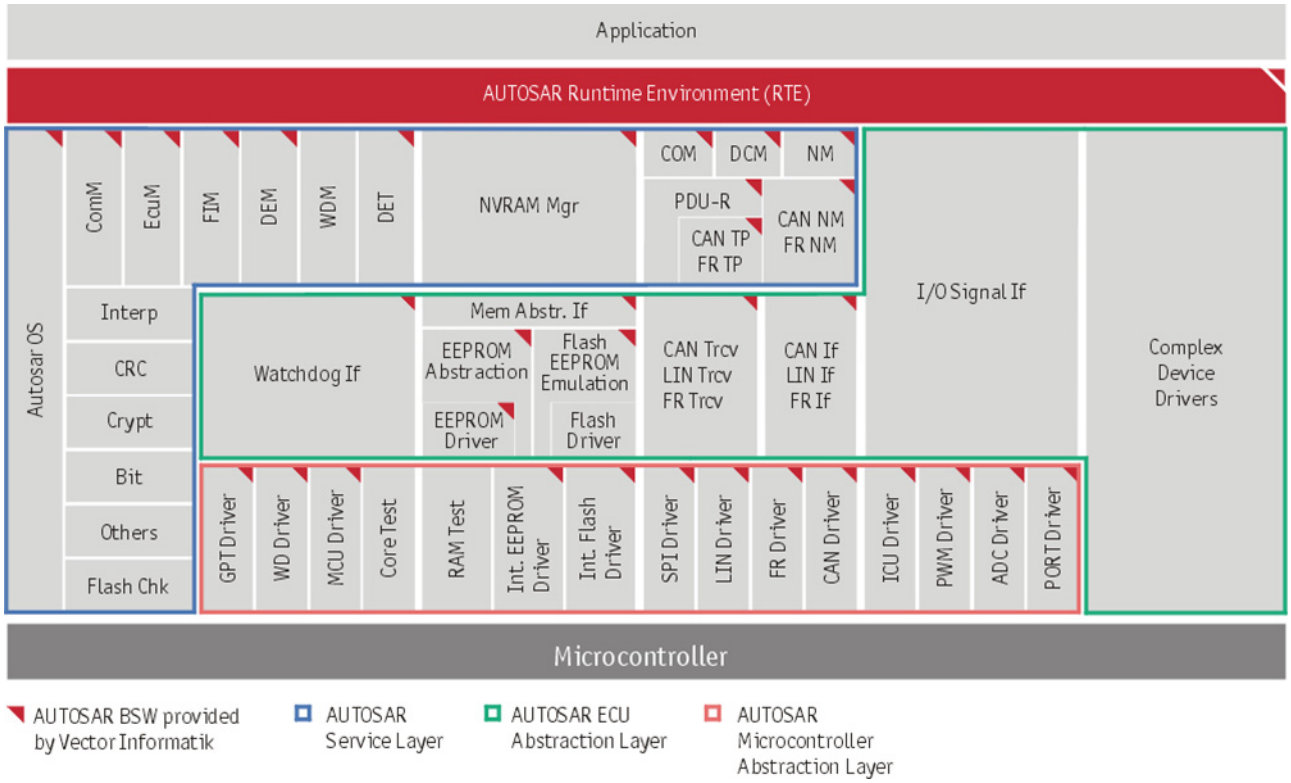

Figure 2-1 AUTOSAR layer model

The FIM is closely related to the Diagnostic Event Manager (DEM), since diagnostic events and their status information are used as inhibit conditions.

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

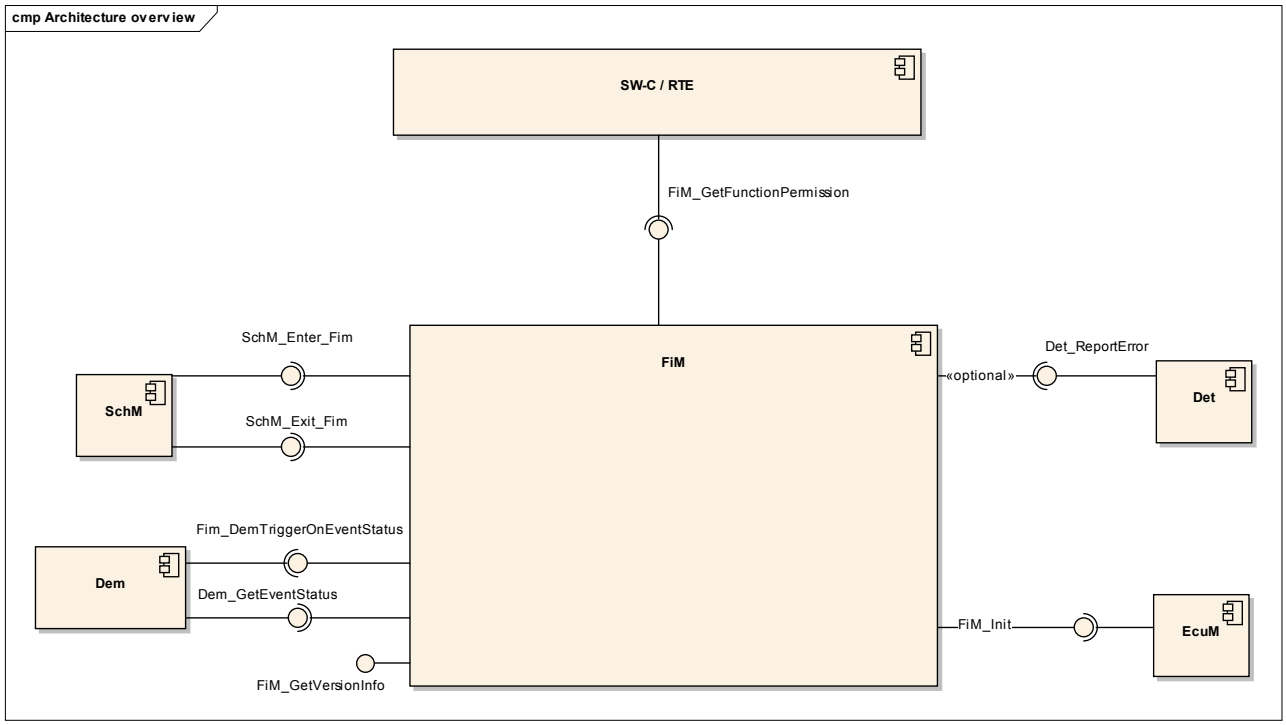

Figure 2-2 Interfaces to adjacent modules of the FiM

Applications do not access the services of the BSW modules directly. They use the service ports provided by the BSW modules via the RTE. The service ports provided by the FiM are listed in chapter 5.4 and are defined in [1].

3 Functional Description

3.1 Features

The features listed in this chapter cover the complete functionality specified in [1].

The "supported" and "not supported" features are presented in the following two tables. For further information of not supported features also see chapter 7.

The following features described in [1] are supported:

Supported Feature
Get function permission by FID
Use DEM Events status as inhibition condition
DEM notify FIM on EventID status change
Pre-compile and link-time configuration
AUTOSAR service component template generation
Development error detection over DET (Except of FIM57)

Table 3-1 Supported SWS features

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

Not Supported Feature
Post build configuration
Cyclic evaluation of DEM Events
Summarized events
Configuration via calibration tool
Evaluation on FID permission status request

Table 3-2 Not supported SWS features

3.2 Initialization

The DEM has to be initialized before the FiM. The FiM calculates all inhibition states at initialization, because it has to use the Service Dem_GetEventStatus of the DEM.

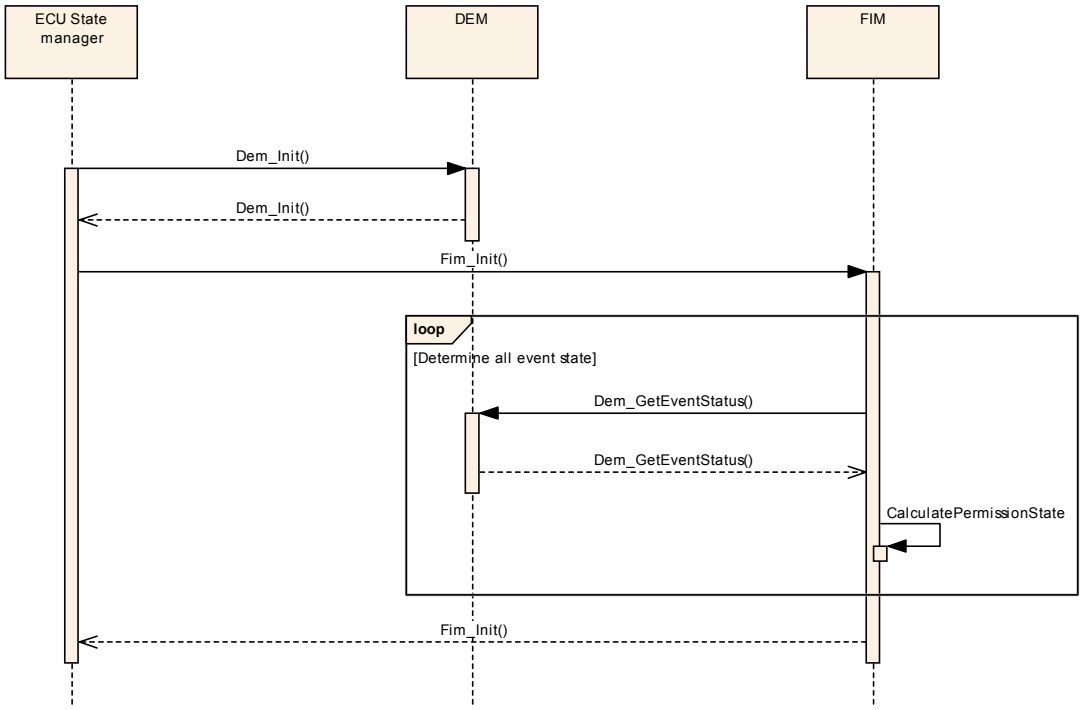

Figure 3-1 Initialization sequence of the FiM

3.3 States

The FiM has no internal states except of initialized and not initialized.

3.4 Main Functions

The actual FiM implementation only supports the Event triggered calculation of the inhibition conditions. Therefore, no main task is to be called,

3.5 Error Handling

3.5.1 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 `FIM_DEV_ERROR_DETECT==STD_ON)`.

If another module is used for development error reporting, the function prototype for reporting the error can be configured by the integrator, but must have the same signature as the service `Det_ReportError()`.

The reported FiM ID is 11.

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	Fim_Init
0x01	Fim_GetFunctionPermission
0x02	Fim_DemTriggerOnEventStatus
0x03	Fim_DemInit
0x04	Fim_GetVersionInfo

Table 3-3 Mapping of service IDs to services

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

Error Code		Description
0x01	FIM_E_WRONG_PERMISSION_REQ	The error code will be used, if the function “Fim_GetFunctionPermission” or “Fim_DemTriggerOnEventStatus” are called before the FiM is initialized.
0x03	FIM_E_FID_OUT_OF_RANGE	The error code will be used, if the function “Fim_GetFunctionPermission” is called with a not configured error code.
0x80	FIM_E_DEM_GETEVENTSTATUS_WRONG_RETURN_VALUE	The error code will be used, if the calculation of the permission states could not be executed during initialization.

Table 3-4 Errors reported to DET

3.5.2 Production Code Error Reporting

Production code related errors are not defined for the FiM.

4 Integration

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

4.1 Scope of Delivery

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

4.1.1 Static Files

File Name	Description
FiM.c	Main source file
FiM.h	Main header file
FiM_Types.h	Header file containing all types.

Table 4-1 Static files-

4.1.2 Dynamic Files

The dynamic files are generated by the configuration tool GENy.

File Name	Description
Fim_Cfg.h	Header file containing definition and extern declaration for FIM_PBcfg.c, Fim.c Fim_Lcfg.h
Fim_PBcfg.c	Source file containing data that is changeable at the pre link time.
Fim_Lcfg.c	Source file containing Post Build data.

Table 4-2 Generated files

4.2 Include Structure

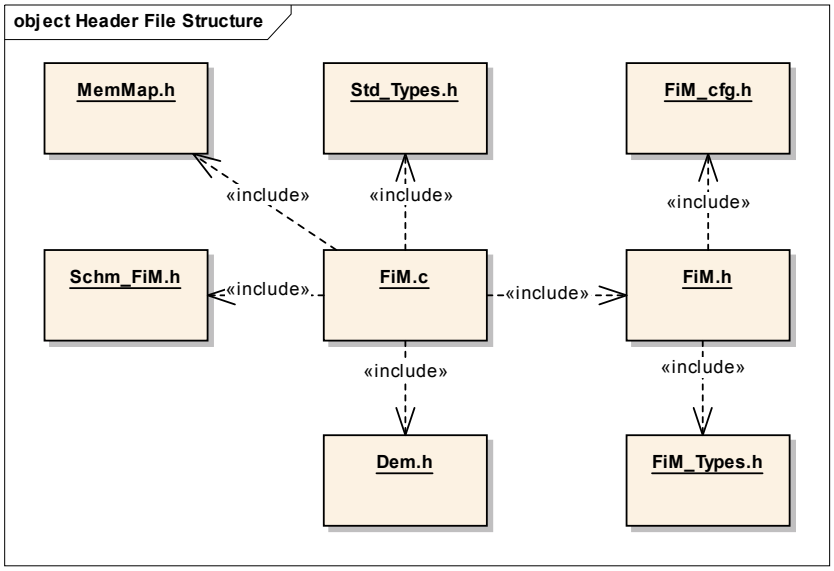

Figure 4-1 Include structure

4.3 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 defined for the FiM and illustrate their assignment among each other.

Compiler Abstraction Definitions / Memory Mapping Sections	FIM_CODE	FIM_CONST	FIM_VAR_NOINIT
FIM_START_SEC_CODE FIM_STOP_SEC_CODE	■		
FIM_START_SEC_CONST_UNSPECIFIED FIM_STOP_SEC_CONST_UNSPECIFIED		■	
FIM_START_SEC_LCFG FIM_STOP_SEC_LCFG		■	
FIM_START_SEC_VAR_NOINIT_8BIT FIM_STOP_SEC_VAR_NOINIT_8BIT			■
FIM_START_SEC_VAR_NOINIT_UNSPECIFIED FIM_START_SEC_VAR_NOINIT_UNSPECIFIED			■

Table 4-3 Compiler abstraction and memory mapping

4.4 Critical Sections

To protect internal data structures against unwanted modification, the FiM uses “Critical Sections” for blocking concurrent access.

`Fim_EnterCritical()` marks the start of a critical part of the control flow, `Fim_LeaveCritical()` marks the end.

Info
It is possible that FiM executes nested internal routines and some or all of them enter into the critical section again (section nesting).

The FiM uses APIs from AUTOSAR Schedule Manager to handle the critical sections (SchM_FiM.h is included).

Caution
You must take special care that the component implementing the critical section (e.g. AUTOSAR Schedule Manager) is already started before the FiM is run.

You have to map the FiM critical sections to the appropriate resource locking method. FiM supports only the `FIM_EXCLUSIVE_AREA_0` and it shall be always mapped to global interrupt disabling, since FiM has always very short time critical sections. The real critical section duration depends on the performance of the controller used in your system, but the FiM critical section design restricts the code to very few instructions.

5 API Description

For an interfaces overview please see Figure 2-2.

5.1 Type Definitions

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

Type Name	C-Type	Description	Value Range
Fim_FunctionIdType	uint16, uint8	Type for the FunctionID	0…255 Size depends on System complexity.
			0..65535 Size depends on System complexity.
Fim_InhibitionConditionType	uint8	Type for the Inhibition Condition	0…255
Fim_EventToFidTableIndexType	uint16	Type to describe the connection from the Event Id Table to the Fid table	0..65535
Fim_FidCounterType	uint8	Type for the calculation of the inhibition counter for each Fid	0…255

Table 5-1 Type definitions

Fim_EventIdTableType

Struct Element Name	C-Type	Description	Value Range
Fim_EventId	Dem_EventIdType	EventId of the DEM	1...255, 1...65535
Fim_EventToFidTableIndex	Fim_EventToFidTableIndexType	Connection from the Event Id Table to the Fid table	0..65535

Table 5-2 Fim_EventIdTableType

Fim_EventToFidTableIndex

Struct Element Name	C-Type	Description	Value Range
Fim_Fid	Fim_FunctionIdType	Configured Function Id	0…255 0..65535 Size depends on System complexity.
Fim_InhibitionCondition	Fim_InhibitionConditionType	Configured Inhibtiion condition for the Fid.	0…255

Table 5-3 Fim_EventIdTableType

5.2 Services provided by FiM

The FiM API consists of services, which are realized by function calls.

5.2.1 Fim_Init

Prototype	
void **Fim_Init** (void)	
Parameter	
-	-
Return code	
ret_code	-
Functional Description	
This function initializes the Function Inhibition Manager and the FIDs	
Particularities and Limitations	
■ Has to be called after the DEM is initialized ■ Must be called before any call of the FIM API	
Expected Caller Context	
-	

Table 5-4 Fim_Init

5.2.2 Fim_DemInit

Prototype	
`void Fim_DemInit (void)`	
Parameter	
-	-
Return code	
-	-
Functional Description	
This function re-initializes the Function Inhibition Manager and must be called by the DEM if the status of certain EventIds changes	
Particularities and Limitations	
■ May only be called once upon startup	
Expected Caller Context	
-	

Table 5-5 Fim_DemInit

5.2.3 Fim_GetFunctionPermission

Prototype	
`void Fim_GetFunctionPermission ( Fim_FunctionIdType FID, Boolean* Permission )` Parameter	
Parameter	
FID	Identification of a functionality by assigned FID. The FID is configured in the FIM.
Permission	TRUE: FID has permission to run FALSE: FID has no permission to run
Return code	
-	-
Functional Description	
API for requesting the permission state to the functionality assigned to the FID. The permission will be set to FALSE, if the FIM is not initialized or if the FID is not valid. If development error reporting is enabled, an error will additionally be reported to the DET (*3.5.1 Development Error Reporting*).	
Particularities and Limitations	
■ FIM has to be initialized	
Expected Caller Context	
-	

Table 5-6 Fim_GetFunctionPermission

5.2.4 Fim_DemTriggerOnEventStatus

Prototype	
`Std_ReturnType` **`Fim_DemTriggerOnEventStatus`** `( Dem_EventIdType EventId, DemStatusExtendedType EventStatusOld, DemStatusExtendedType EventStatusNew )`	
Parameter	
EventId	Identification of an Event by the assigned event number. The Event Number is configured in the DEM.
EventStatusOld	Extended event status before change
EventStatusNew	Extended event status after change
Return code	
Std_ReturnType	Returns always E_OK
Functional Description	
This function has to be called by the DEM if the status of an event changed	
Particularities and Limitations	
■ FIM has to be initialized	
Expected Caller Context	
-	

Table 5-7 Fim_GetFunctionPermission

5.2.5 Fim_GetVersionInfo

Prototype	
`void Fim_GetVersionInfo ( Std_VersionInfoType *versioninfo)`	
Parameter	
versioninfo	Pointer to the structure containing the version information
Return code	
-	-
Functional Description	
Function to acquire version information	
Particularities and Limitations	
■ The function is only available if enabled at compile time (FIM_VERSION_INFO_API = ON)	
Expected Caller Context	
-	

Table 5-8 Fim_GetVersionInfo

5.3 Services used by the Function Inhibition Manager

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

Component	API
DET	Det_ReportError
DEM	Dem_GetEventStatus

Table 5-9 Services used by the FiM

5.4 Service Ports

5.4.1 Client Server Interface

A client server interface is related to a Provide Port at the server side and a Require Port at client side.

5.4.1.1 Provide Ports on FiM Side

At the Provide Ports of the FiM the API functions described in 5.2 are available as Runnable Entities. The Runnable Entities are invoked via Operations. The mapping from a SWC client call to an Operation is performed by the RTE. In this mapping the RTE adds Port Defined Argument Values to the client call of the SWC, if configured.

The following sub-chapters present the Provide Ports defined for the FiM and the Operations defined for the Provide Ports, the API functions related to the Operations and the Port Defined Argument Values to be added by the RTE.

5.4.1.1.1 FunctionInhibition

Operation	API Function	Port Defined Argument Values
FunctionInhibition	Fim_GetFunctionPermission	FunctionIdType FunctionId

6 Configuration

In the FiM the attributes can be configured in GENy.

6.1 Configuration with GENy

The FiM is configured with the help of the configuration tool GENy. …

The following chapter explains the properties of the Function Inhibition Manager which can be configured in the generator.

6.1.1 Pre-compile Properties

The following figure shows a list of all pre-compile options in the FIM.

Configurable Options	SysService_AsrFim
– SysService_AsrFim	
– General	
Cyclic Event Evaluation	☐ *
Data Fixed	☑ *
Development Error Detection	☑
Get Version Info	☑
– Software Component Template	
Software Component Name	Fim*
– User Config Files	
User Config File	* ...
– Functions	
Add new Function	...

Figure 6-1 GENy pre-compile configuration

Parameter	Description
Cyclic Event Evaluation	This configuration parameter specifies whether the evaluation of DEM events is performed by the FIM cyclically or the DEM informs FIM about changes of event states. ON: FIM cyclically evaluates event states in the DEM. OFF: DEM informs FIM about changes of event states. In this version it's not possible to change this option. Only the notification by DEM is implemented.
Data Fixed	If this parameter is selected the inhibition conditions can't be changed by post-build configuration. In this version it's not possible to change this option. No post-build possibility is implemented.
Development Error Detection	If 'Development Error Detection' is enabled, development errors are reported to the Development Error Tracer (DET). The following development errors may occur within the FIM: ○ FIM_E_WRONG_PERMISSION_REQ occurs if API Fim_GetPermissionStatus was called before complete initialization of Fim ○ FIM_E_WRONG_TRIGGER_ON_EVENT occurs if DEM calls FIM before it is completely initialized ○ FIM_E_FID_OUT_OF_RANGE occurs if API Fim_GetPermissionStatus was called with wrong FID ○ FIM_E_EVENTID_OUT_OF_RANGE occurs if DEM calls FIM with wrong EventId ○ FIM_E_DEM_GETEVENTSTATUS_WRONG_RETURN_VALUE occurs if DEM API Dem_GetEventStatus returns not E_OK All development errors are reported to the Development Error Tracer. Note: In general, the development error detection is recommended during pre-test phase. It is not recommended to enable the development error detection in production code due to increased runtime and ROM needs.
Version Info API	En-/ disable the function Fim_GetVersionInfo() to get the major, minor and patch version information.
Software Component Name	This name is used as component name (ShortName) in the generated software component template. Change this name if you try to import the FIM components of several ECUs and have problems with name clashes.
User Config File	A configuration file is generated by GENy. If you want to overwrite settings in the generated configuration file, you can specify a path to a user defined configuration file. The user defined configuration file will be included at the end of the generated file. Therefore definitions in the user defined configuration file can overwrite definitions in the generated configuration file.

Add new function	With this button it is possible to add another function ID.

Table 6-1 GENy pre-compile configuration

6.1.2 FunctionID specific Properties

By a right-mouse click on the component 'SysService_AsrFim' in the component tree a context menu is opened which offers to add a Function ID (FID) ('Add Function ID'). A Function ID can also be added with the button “Add new function” in the FIM configuration page. With each added FID another context menu can be opened which offers to add a new or remove the selected FID ('Add Function ID' and 'Remove Function'). For each FID the inhibition condition can be specified by adding one or more DEM events and an inhibition condition for each of these DEM events. The status of a DEM event is evaluated by comparing the inhibition condition with the event state in the DEM. If at least one DEM event status fulfils the inhibition condition, the FID state is set to 'Inhibited'.

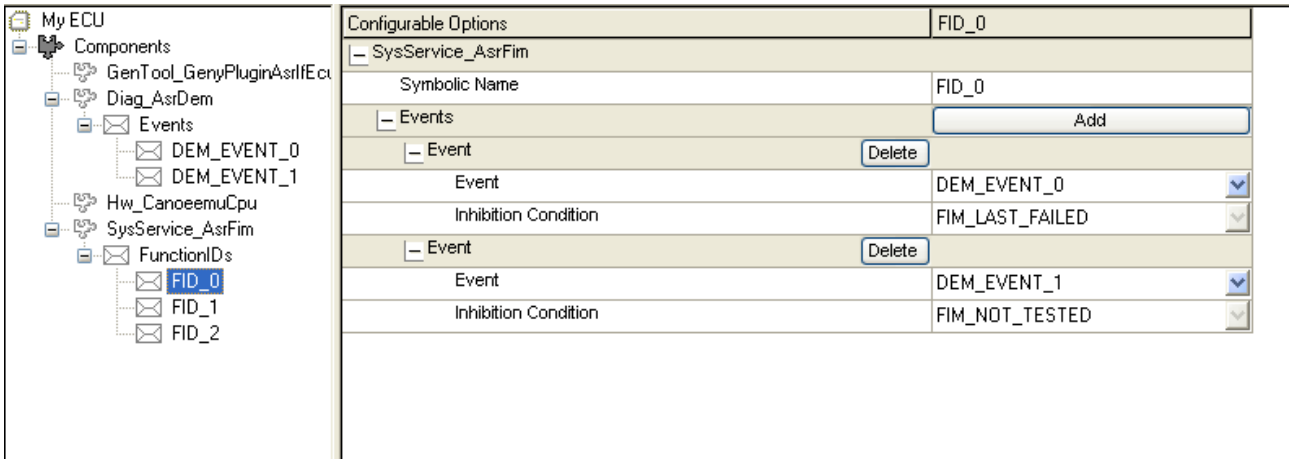

Figure 6-2 GENy FID specific configuration

Parameter	Description
Symbolic Name	The name of the FunctionID
Event	You can select a DEM event to specify the inhibition condition for the Function ID (FID). The status of each DEM event is evaluated by comparing the inhibition condition with the event state in the DEM. If at least one DEM event status fulfils the inhibition condition, the FID state is set to 'Inhibited'.
Inhibition Condition	You can select an inhibition condition for the corresponding DEM event. If the inhibition condition matches the DEM state of the DEM event, the event evaluation fails and the FID state becomes 'Inhibited'.

Table 6-2 GENy FID specific configuration

7 AUTOSAR Standard Compliance

7.1 Deviations

See *Table 3-2 Not supported SWS features.*

7.2 Additions/ Extensions

No Additions and Extensions are implemented.

7.3 Limitations

Limitations are not known.

8 Glossary and Abbreviations

8.1 Glossary

Term	Description
EAD	Embedded Architecture Designer; generation tool for MICROSAR components
GENy	Generation tool for CANbedded and MICROSAR components

Table 8-1 Glossary

8.2 Abbreviations

Abbreviation	Description
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
BSW	Basis Software
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 8-2 Abbreviations

9 Contact

Visit our website for more information on

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

www.vector-informatik.com