

# MICROSAR ETHFW

## Technical Reference

Ethernet Firewall

Version 1.0.0

Authors	Philipp Christmann
Status	Released

## Document Information

### History

Author	Date	Version	Remarks
Philipp Christmann	2016-12-02	1.0.0	Creation of document

### Reference Documents

No.	Source	Title	Version
[1]	AUTOSAR	AUTOSAR_SWS_EthernetInterface.pdf	4.2.2
[2]	AUTOSAR	AUTOSAR_SWS_DefaultErrorTracer.pdf	4.2.2
[3]	AUTOSAR	AUTOSAR_TR_BSWModuleList.pdf	4.2.2

### Scope of the Document

This technical reference describes the general use of the Ethernet Firewall (ETHFW) basis software module.



#### 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

<b>1</b>	<b>Component History .....</b>	<b>6</b>
<b>2</b>	<b>Introduction.....</b>	<b>7</b>
2.1	Architecture Overview .....	8
<b>3</b>	<b>Functional Description .....</b>	<b>10</b>
3.1	Features .....	10
3.2	Initialization .....	10
3.2.1	Configuration Variants 1, 2 (Pre-Compile and Link-Time).....	10
3.2.2	Configuration Variant 3 (Post-build).....	10
3.3	States .....	11
3.4	Main Functions .....	11
3.5	Interaction with ETHIF .....	11
3.5.1	Frame Reception .....	11
3.5.2	Frame Transmission.....	12
3.6	Error Handling.....	13
3.6.1	Development Error Reporting.....	13
<b>4</b>	<b>Integration.....</b>	<b>15</b>
4.1	Scope of Delivery.....	15
4.1.1	Static Files .....	15
4.1.2	Dynamic Files .....	15
4.2	Critical Sections .....	15
<b>5</b>	<b>API Description.....</b>	<b>16</b>
5.1	Type Definitions .....	16
5.2	Services provided by ETHFW .....	17
5.2.1	EthFw_InitMemory .....	17
5.2.2	EthFw_Init.....	18
5.3	Services used by ETHFW .....	19
5.4	Callback Functions.....	20
5.4.1	EthFw_IsFrameRxAllowed .....	20
5.4.2	EthFw_IsFrameTxAllowed .....	21
5.5	Configurable Interfaces.....	22
5.5.1	EthFw_GetVersionInfo .....	22
5.5.2	Notifications .....	23
5.5.2.1	<RxFrameDiscardedCbk> .....	23
5.5.2.2	<TxFrameDiscardedCbk> .....	24

<b>6</b>	<b>Configuration.....</b>	<b>25</b>
6.1	Configuration Variants.....	25
6.2	Configuration with DaVinci Configurator Pro .....	25
6.2.1	Configuration of EthFwGeneral .....	25
6.2.2	Configuration of optional User-Callouts .....	26
6.2.3	Configuration of EthFwRule .....	26
6.2.4	Configuration of Value-Ranges.....	26
<b>7</b>	<b>Glossary and Abbreviations .....</b>	<b>28</b>
7.1	Glossary .....	28
7.2	Abbreviations .....	28
<b>8</b>	<b>Contact.....</b>	<b>29</b>

## Illustrations

Figure 2-1	AUTOSAR 4.2 Architecture Overview .....	8
Figure 2-2	Interfaces to adjacent modules of the ETHFW .....	9
Figure 3-1	Frame reception .....	12
Figure 3-2	Frame transmission .....	13
Figure 6-1	EthFwGeneral container .....	25
Figure 6-2	Structure of EthFwRule configuration .....	26
Figure 6-3	Configuration of local and remote values .....	27

## Tables

Table 1-1	Component history.....	6
Table 3-1	Supported Features .....	10
Table 3-2	Feature Limitations .....	10
Table 3-3	Service IDs .....	14
Table 3-4	Errors reported to DET .....	14
Table 4-1	Static files .....	15
Table 4-2	Generated files .....	15
Table 5-1	EthIf_FrameHdrType .....	16
Table 5-2	EthFw_InitMemory .....	17
Table 5-3	EthFw_Init .....	18
Table 5-4	Services used by the ETHFW .....	19
Table 5-5	EthFw_IsFrameRxAllowed.....	20
Table 5-6	EthFw_IsFrameTxAllowed .....	21
Table 5-7	EthFw_GetVersionInfo.....	22
Table 5-8	<RxFrameDiscardedCbk> .....	23
Table 5-9	<TxFrameDiscardedCbk> .....	24
Table 7-1	Glossary .....	28
Table 7-2	Abbreviations.....	28

## 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.xx	Created Beta version

Table 1-1 Component history

## 2 Introduction

This document describes the functionality, API and configuration of the BSW module ETHFW. The Ethernet Firewall is implemented as an UpperLayer of the ETHIF module specified in [1].

<b>Supported AUTOSAR Release*:</b>	4.2.2	
<b>Supported Configuration Variants:</b>	pre-compile	
<b>Vendor ID:</b>	ETHFW_VENDOR_ID	30 decimal (= Vector-Informatik, according to HIS)
<b>Module ID:</b>	ETHFW_MODULE_ID	255 decimal (according to ref. [3] )

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

ETHFW provides an implementation of a Firewall for Ethernet communication. The main task is it to block unwanted traffic which is received or transmitted in order to increase the security of the entire network.

Therefore, the ETHIF module is extended by additional callouts to the ETHFW module which allow the firewall module to check the current frame and to discard it if the frame does not match the configured set of rules.

The ETHFW module allows it to specify filter rules for different types of traffic (IPv4/IPv6, AVB, RAW-Ethernet) as well as on different layers (UDP, TCP, RAW).

## 2.1 Architecture Overview

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

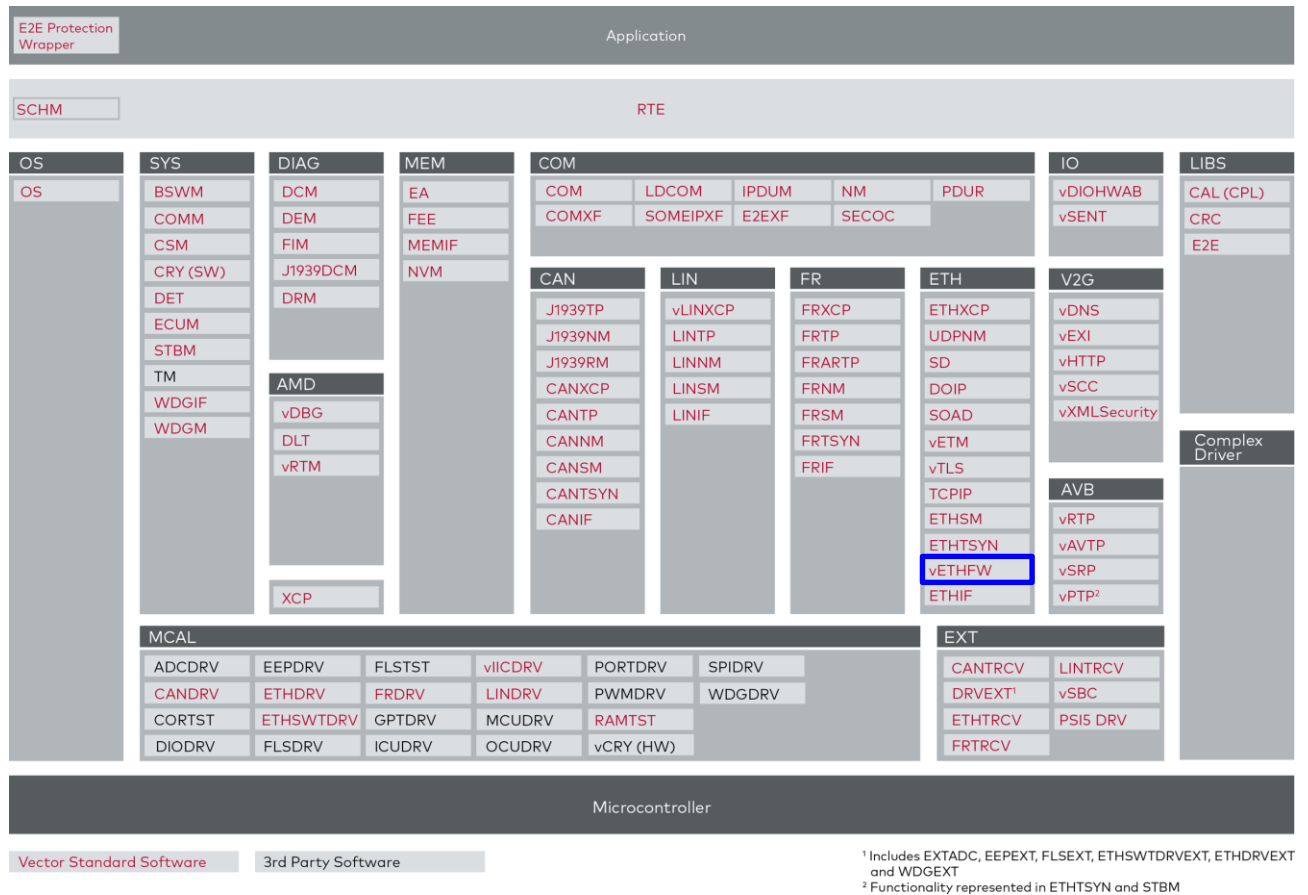


Figure 2-1 AUTOSAR 4.2 Architecture Overview



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

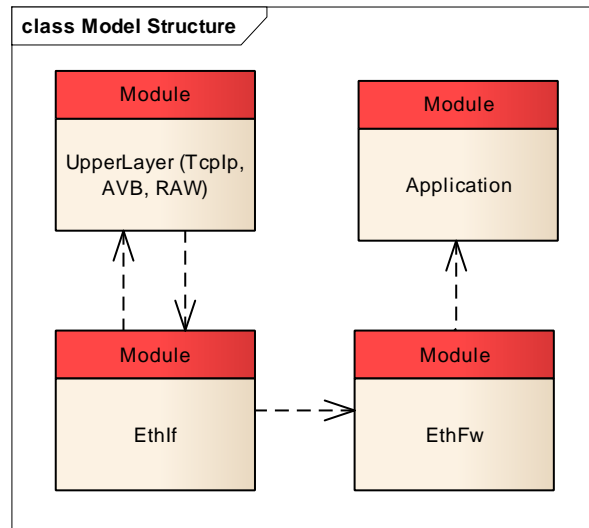


Figure 2-2 Interfaces to adjacent modules of the ETHFW

## 3 Functional Description

### 3.1 Features

The features listed in the following table cover the complete functionality specified for the ETHFW.

- ▶ Table 3-1 Supported Features
- ▶ Table 3-2 Feature Limitations

The following features are supported:

Supported Features
Implemented as DENY-ALL firewall.
Post-build loadable support.
Filter based on Ethernet information. (VLAN, frame priority, EtherType, MAC addresses, next layer protocol)
Filter based on AVB information. (Stream ID)
Filter based on IP information. (IP addresses, next layer protocol)
Filter based on IP protocol. (UDP, TCP, RAW)
Filter based on UDP/TCP protocol. (Ports)

Table 3-1 Supported Features

The following features are not supported:

Feature Limitations
Stateful Packet Inception (SPI)
Intrusion Detection System (IDS)

Table 3-2 Feature Limitations

### 3.2 Initialization

The ETHFW module is initialized via a `EthFw_InitMemory()` call followed by a call of `EthFw_Init()`.

#### 3.2.1 Configuration Variants 1, 2 (Pre-Compile and Link-Time)

At configuration Variant 1 (Pre-compile) and Variant 2 (Link-Time) the pointer given to `EthFw_Init()` is ignored. The passed pointer has to be a `NULL_PTR`. At these configuration variants the ETHFW module has direct access to all configuration data.

#### 3.2.2 Configuration Variant 3 (Post-build)

In this configuration Variant, the ETHFW module has to be initialized using the `EthFw_Init()` function with the address of the post-build configuration data passed as parameter. The declaration of the post-build configuration data is contained in the files `EthFw_PBcfg.h` and `EthFw_PBcfg.c`.

### 3.3 States

The ETHFW module is stateless and does not provide different states.

### 3.4 Main Functions

The ETHFW module is only triggered on reception or transmission of Ethernet messages and does not implement a cyclically called MainFunction.

### 3.5 Interaction with ETHIF

The firewall module is integrated to the Ethernet stack by callouts triggered by the ETHIF module. On each message reception or transmission, the Ethernet Interface module passes the frame to the ETHFW module in order to check the compliance with the configured firewall rules. If the frame does not correspond with the defined rules, the ETHIF is requested to discard the message.

#### 3.5.1 Frame Reception

Figure 3-1 depicts the interaction during frame reception. The result of the firewall checks can also be forwarded by an optional callout function. The callout can be activated and configured with the BSWMD parameter `EthFwRxFrameDiscardedCbk`.

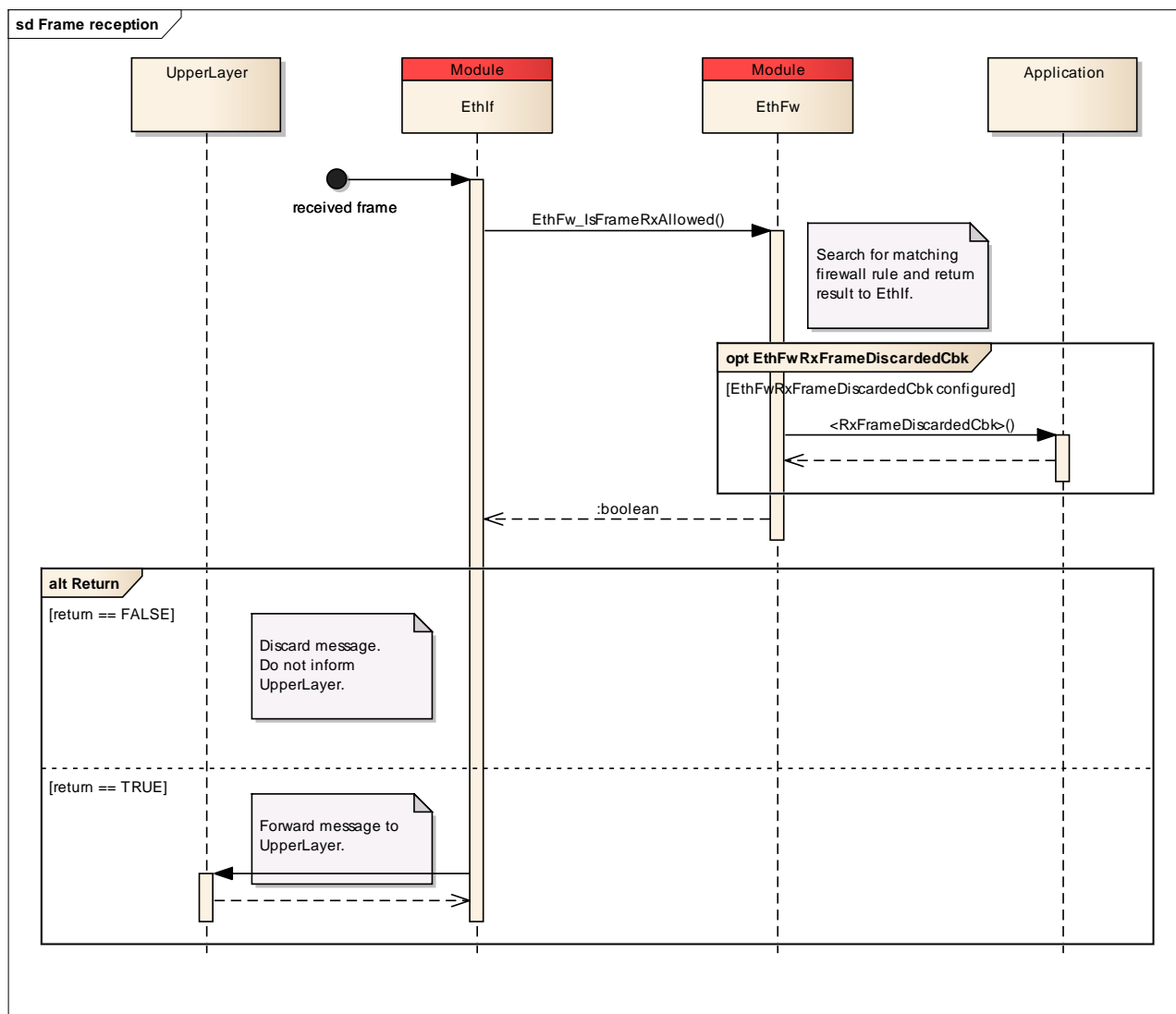


Figure 3-1 Frame reception

### 3.5.2 Frame Transmission

Figure 3-2 depicts the interaction during frame transmission. The result of the firewall checks can also be forwarded by an optional callout function. The callout can be activated and configured with the BSWMD parameter `EthFwTxFrameDiscardedCbk`.

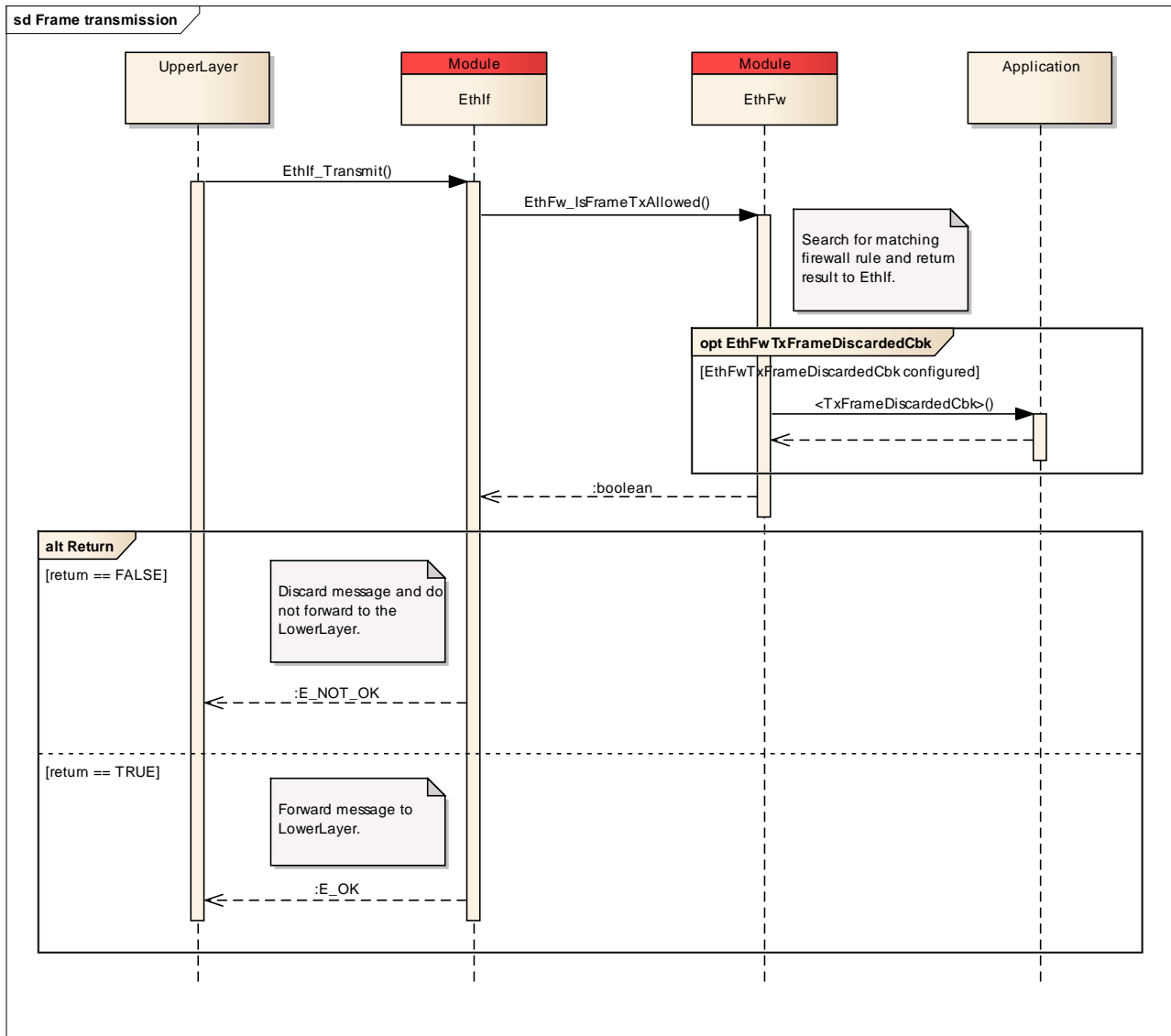


Figure 3-2 Frame transmission

## 3.6 Error Handling

### 3.6.1 Development Error Reporting

By default, development errors are reported to the DET using the service `Det_ReportError()` as specified in [2], if development error reporting is enabled (i.e. pre-compile parameter `ETHFW_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 ETHFW ID is 255.

The reported service IDs identify the services which are described in 0. The following table presents the service IDs and the related services:

Service ID	Service
ETHFW_SID_IS_RX_FRAME_ALLOWED	EthFw_IsFrameRxAllowed()
ETHFW_SID_IS_TX_FRAME_ALLOWED	EthFw_IsFrameTxAllowed()

Table 3-3 Service IDs

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

Error Code	Description
ETHFW_E_NO_ERROR	No error occurred.
ETHFW_E_PARAM_CONFIG	Wrong initialization pointer passed to initialization function.
ETHFW_E_PARAM_POINTER	Invalid pointer in API call.
ETHFW_E_UNINIT	Module is not initialized.
ETHFW_E_ALREADY_INITIALIZED	Module was already initialized.
ETHFW_E_INIT_FAILED	Initialization of the module failed.

Table 3-4 Errors reported to DET

## 4 Integration

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

### 4.1 Scope of Delivery

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

#### 4.1.1 Static Files

File Name	Description
EthFw.c	This is the source file of the ETHFW module.
EthFw.h	API declaration of the module.
EthFw_Cbk.h	API declaration of ETHFW callback functions.
EthFw_Priv.h	Component local macro and variable declaration.

Table 4-1 Static files

#### 4.1.2 Dynamic Files

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

File Name	Description
EthFw_Cfg.h	Pre-compile time parameter configuration.
EthFw_Lcfg.c	Link-time parameter configuration.
EthFw_Lcfg.h	Link-time parameter configuration declaration.
EthFw_PBcfg.c	Post-build parameter configuration.
EthFw_PBcfg.h	Post-build parameter configuration declaration.

Table 4-2 Generated files

### 4.2 Critical Sections

To ensure data consistency and a correct function of the ETHFW module an exclusive area is used and has to be provided during the integration.

Considering the timing behavior of your system (e.g. depending on the CPU load of your system, priorities and interruptibility of interrupts and OS tasks and their jitter and delay times) the integrator has to choose and configure a critical section solution in such way that it is ensured that the API functions do not interrupt each other.

It is recommended to use the functions `SuspendAllInterrupts()` and `ResumeAllInterrupts()` to ensure data consistency.

## 5 API Description

For an interfaces overview please see Figure 2-2.

### 5.1 Type Definitions

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

#### EthIf\_FrameHdrType

This structure contains information about the discarded Ethernet frame.

Struct Element Name	C-Type	Description
DstMacAddrPtr	const uint8 *	Pointer to destination MAC address (network byte order)
SrcMacAddrPtr	const uint8 *	Pointer to source MAC address (network byte order)
FrameType	Eth_FrameType (uint16)	Ethernet frame type
VlanId	uint16	Ethernet frame VLAN ID (#define ETHIF_INV_VLAN_ID 0xFFF)
Priority	uint8	Ethernet frame priority

Table 5-1 EthIf\_FrameHdrType



## 5.2 Services provided by ETHFW

### 5.2.1 EthFw\_InitMemory

Prototype	
void <b>EthFw_InitMemory</b> (void)	
Parameter	
none	
Return code	
void	void
Functional Description	
Function for EthFw-variable initialization.	
Particularities and Limitations	
<p>Module is uninitialized.</p> <p>Service to initialize module global variables at power up. This function initializes the variables in EthFw sections. Used in case they are not initialized by the startup code.</p>	
Call context	
<ul style="list-style-type: none"> <li>&gt; TASK</li> <li>&gt; This function is Synchronous</li> <li>&gt; This function is Non-Reentrant</li> </ul>	

Table 5-2 EthFw\_InitMemory

## 5.2.2 EthFw\_Init

Prototype	
<pre>void <b>EthFw_Init</b> (     const EthFw_ConfigType *ConfigPtr)</pre>	
Parameter	
ConfigPtr [in]	Configuration structure for initializing the module.
Return code	
void	void
Functional Description	
Initialization function.	
Particularities and Limitations	
<p>Specification of module initialization</p> <ul style="list-style-type: none"> <li>&gt; Interrupts are disabled. Module is uninitialized. EthFw_InitMemory has been called unless EthFw_ModuleInitialized is initialized by start-up code.</li> </ul> <p>This function initializes the module EthFw. It initializes all variables and sets the module state to initialized.</p>	
Call context	
<ul style="list-style-type: none"> <li>&gt; TASK</li> <li>&gt; This function is Synchronous</li> <li>&gt; This function is Non-Reentrant</li> </ul>	

Table 5-3 EthFw\_Init

### 5.3 Services used by ETHFW

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

Component	API
DET	Det_ReportError

Table 5-4 Services used by the ETHFW

## 5.4 Callback Functions

This chapter describes the callback functions that are implemented by the ETHFW and can be invoked by other modules. The prototypes of the callback functions are provided in the header file `EthFw_Cbk.h`.

### 5.4.1 EthFw\_IsFrameRxAllowed

Prototype	
<pre>boolean EthFw_IsFrameRxAllowed (     uint8 CtrlIdx,     const EthIf_FrameHdrType *FrameHdrPtr,     const uint8 *PayloadPtr,     uint16 PayloadLen)</pre>	
Parameter	
CtrlIdx [in]	Index of the Ethernet controller.
FrameHdrPtr [in]	Structure containing Ethernet header information
PayloadPtr [in]	Pointer to payload which was received and shall be forwarded
PayloadLen [in]	Payload length
Return code	
boolean	TRUE - Forward message
boolean	FALSE - Discard message
Functional Description	
Checks if the message matches the configured filter rules and shall be received.	
Particularities and Limitations	
Module has been initialized	
Call context	
<ul style="list-style-type: none"> <li>&gt; ANY</li> <li>&gt; This function is Synchronous</li> <li>&gt; This function is Reentrant</li> </ul>	

Table 5-5 EthFw\_IsFrameRxAllowed

## 5.4.2 EthFw\_IsFrameTxAllowed

Prototype	
<pre>boolean EthFw_IsFrameTxAllowed (     uint8 CtrlIdx,     const EthIf_FrameHdrType *FrameHdrPtr,     const uint8 *PayloadPtr,     uint16 PayloadLen)</pre>	
Parameter	
CtrlIdx [in]	Index of the Ethernet controller.
FrameHdrPtr [in]	Structure containing Ethernet header information
PayloadPtr [in]	Pointer to payload which shall be transmitted
PayloadLen [in]	Payload length
Return code	
boolean	TRUE - Transmit message
boolean	FALSE - Discard message
Functional Description	
Checks if the message matches the configured filter rules and shall be transmitted.	
Particularities and Limitations	
Module has been initialized	
Call context	
<ul style="list-style-type: none"><li>&gt; ANY</li><li>&gt; This function is Synchronous</li><li>&gt; This function is Reentrant</li></ul>	

Table 5-6 EthFw\_IsFrameTxAllowed

## 5.5 Configurable Interfaces

### 5.5.1 EthFw\_GetVersionInfo

Prototype	
<pre>void <b>EthFw_GetVersionInfo</b> (     Std_VersionInfoType *VersionInfoPtr)</pre>	
Parameter	
VersionInfoPtr [out]	Pointer to where to store the version information. Parameter must not be NULL.
Return code	
void	void
Functional Description	
Returns the version information.	
Particularities and Limitations	
none EthFw_GetVersionInfo() returns version information, vendor ID and AUTOSAR module ID of the component.	
Call context	
<ul style="list-style-type: none"><li>&gt; TASK ISR2</li><li>&gt; This function is Synchronous</li><li>&gt; This function is Reentrant</li></ul>	

Table 5-7 EthFw\_GetVersionInfo

## 5.5.2 Notifications

At its configurable interfaces the ETHFW defines notifications that can be mapped to callback functions provided by other modules. The mapping is not statically defined by the ETHFW but can be performed at configuration time. The function prototypes that can be used for the configuration have to match the appropriate function prototype signatures, which are described in the following sub-chapters.

### 5.5.2.1 <RxFrameDiscardedCbk>

Prototype	
<pre>void &lt;RxFrameDiscardedCbk&gt; (     uint8 CtrlIdx,     const EthIf_FrameHdrType *FrameHdrPtr,     const uint8 *PayloadPtr,     uint16 PayloadLen)</pre>	
Parameter	
CtrlIdx [in]	Index of the Ethernet controller.
FrameHdrPtr [in]	Structure containing Ethernet header information
PayloadPtr [in]	Pointer to payload which was discarded
PayloadLen [in]	Payload length
Return code	
void	none
Functional Description	
Informs the UpperLayer that a received frame has been discarded and was not forwarded.	
Particularities and Limitations	
Call context	
<ul style="list-style-type: none"> <li>&gt; ANY</li> <li>&gt; This function is Synchronous</li> <li>&gt; This function is Reentrant</li> </ul>	

Table 5-8 <RxFrameDiscardedCbk>

### 5.5.2.2 <TxFrameDiscardedCbk>

Prototype	
<pre>void &lt;TxFrameDiscardedCbk&gt; (     uint8 CtrlIdx,     const EthIf_FrameHdrType *FrameHdrPtr,     const uint8 *PayloadPtr,     uint16 PayloadLen)</pre>	
Parameter	
CtrlIdx [in]	Index of the Ethernet controller.
FrameHdrPtr [in]	Structure containing Ethernet header information
PayloadPtr [in]	Pointer to payload which was discarded
PayloadLen [in]	Payload length
Return code	
void	none
Functional Description	
Informs the UpperLayer that a frame has been discarded and was not transmitted.	
Particularities and Limitations	
Call context	
<ul style="list-style-type: none"><li>&gt; ANY</li><li>&gt; This function is Synchronous</li><li>&gt; This function is Reentrant</li></ul>	

Table 5-9 &lt;TxFrameDiscardedCbk&gt;



## 6 Configuration

In the ETHFW the attributes can be configured with the tool DaVinci Configurator Pro.

### 6.1 Configuration Variants

The ETHFW supports the configuration variants

- ▶ VARIANT-PRE-COMPILE
- ▶ VARIANT-POST-BUILD-LOADABLE

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

### 6.2 Configuration with DaVinci Configurator Pro

The Ethernet firewall module is configured with the help of the configuration tool Configurator Pro. The configuration tool GENy is not supported by this MICROSAR version.

In the following sub-chapters some configuration hints are given to understand how to configure the ETHFW correctly.



#### Introduction

The Ethernet firewall module ETHFW is implemented as DENY-ALL firewall. For each permitted type of Ethernet traffic, a corresponding firewall rule has to be defined.

#### 6.2.1 Configuration of EthFwGeneral

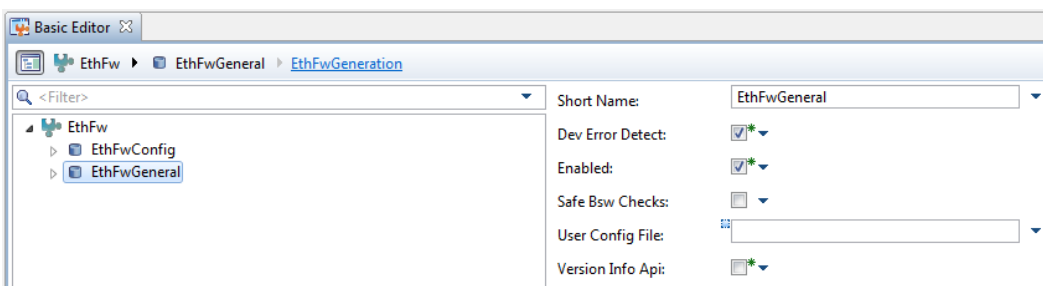


Figure 6-1 EthFwGeneral container

The `EthFwGeneral` container shown in Figure 6-1 contains configuration parameters which are rule-independent and valid for the entire ETHFW module. A detailed description of each configuration parameter can be found in the description view of the parameter properties.

The `EthFwEnabled` switch allows it to deactivate the ETHFW module in order to allow an integration of the entire ECU project without considering firewall rules from the beginning. The resulting validation error message (ETHFW02032) can be ignored temporary for this special, integration scenario.

The `EthFwUserConfigFile` allows the user to define additional code, which will be generated to the end of the `EthFw_Cfg.h` file.

The configuration parameters in the `EthFwGeneration` container control the generation of dynamic files and configure for example additional out of bounds read/write sanitizer or several data reduction strategies.

### 6.2.2 Configuration of optional User-Callouts

The optional parameters `EthFwRxFramDiscardedCbK` and `EthFwRxFramDiscardedCbK` allow the configuration of callout functions which are called by the ETHFW module if a message is rejected by the firewall. These callbacks allow it to detect the rejection as well as to provide the possibility of additional analysis or logging in the application. Therefore the message header as well as the entire payload is forwarded in the callout.

### 6.2.3 Configuration of EthFwRule

All firewall rules are configured as `EthFwRule` which are interpreted as WHITELIST rules. Hence, all traffic which does not match all criteria of the rule is discarded.

The entry point of each rule is the configuration of the relevant tuple of ETHIF frame owner and controller (`EthFwFrameOwnerRef`, `EthFwIfControllerRef`). These references are used to indicate the VLAN ID as well as the Ethernet Frame Type the rule is relevant for.

The remaining configuration parameters are structured hierarchical according the order of protocol headers in the frame as depicted in Figure 6-2. The possible configuration variants considering the multiplicities, values and ranges can be detected in the configuration tool.

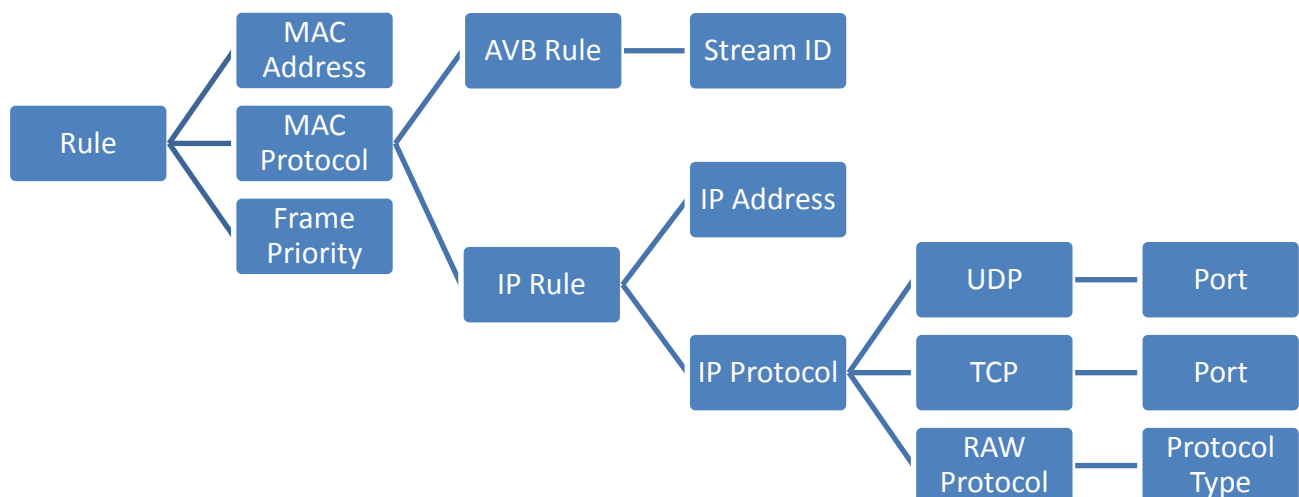
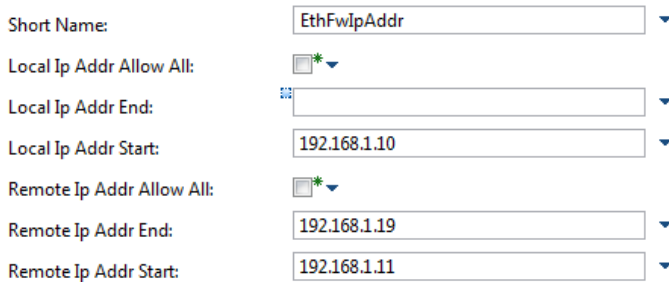


Figure 6-2 Structure of `EthFwRule` configuration

### 6.2.4 Configuration of Value-Ranges

The configuration of MAC and IP addresses as well as ports use the identical kind of configuration which allows the configuration of individual value ranges for the local and

remote value. The usage of “local” and “remote” abstracts and simplifies the typically used terms of “source” and “destination”, because the direction of the message has not to be considered.



Short Name:	EthFwIpAddr
Local Ip Addr Allow All:	<input checked="" type="checkbox"/>
Local Ip Addr End:	
Local Ip Addr Start:	192.168.1.10
Remote Ip Addr Allow All:	<input checked="" type="checkbox"/>
Remote Ip Addr End:	192.168.1.19
Remote Ip Addr Start:	192.168.1.11

Figure 6-3 Configuration of local and remote values

A possible IP Address configuration is depicted in Figure 6-3. There exist three different possibilities to specify an address/port respectively an address/port range:

1. Configuration of START value.  
If only the START value is configured and no END value, a single address/port can be specified. (Figure 6-3: Local IP Address)
2. Configuration of START and END value.  
If both values are configured, an address/port range can be specified. (Figure 6-3: Remote IP Address)
3. Enables ALLOW\_ALL switch.  
If the ALLOW\_ALL switch is enabled, the firewall does not filter on the corresponding address/port and allows all values.

## 7 Glossary and Abbreviations

### 7.1 Glossary

Term	Description
EAD	Embedded Architecture Designer; generation tool for MICROSAR components

Table 7-1 Glossary

### 7.2 Abbreviations

Abbreviation	Description
API	Application Programming Interface
AUTOSAR	Automotive Open System Architecture
BSW	Basis Software
DET	Development Error Tracer
MICROSAR	Microcontroller Open System Architecture (the Vector AUTOSAR solution)
ETHIF	Ethernet Interface
ETHFW	Ethernet Firewall

Table 7-2 Abbreviations

## 8 Contact

Visit our website for more information on

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

[www.vector.com](http://www.vector.com)