



Programming manual

**CR710S**

**CR711S**

**CR720S**

**CR721S**

Operating system: V3.1.x.y or higher

CODESYS: V3.5 SP11

English

# Contents

<b>1</b>	<b>About this manual</b>	<b>7</b>
1.1	Legal and copyright information .....	7
1.2	Purpose of the document .....	7
1.3	Symbols used .....	8
1.4	Warnings used .....	8
1.5	Overview: User documentation for CR7xxS .....	9
1.6	Overview: documentation for CODESYS 3.n .....	9
1.7	CODESYS .....	10
1.8	Change history .....	11
<b>2</b>	<b>Safety instructions</b>	<b>14</b>
2.1	Required previous knowledge .....	14
2.2	Important standards .....	14
2.3	Note! .....	15
2.4	Start-up behaviour of the controller .....	16
2.5	IT safety .....	16
2.6	Access protection .....	16
<b>3</b>	<b>Functions and features</b>	<b>18</b>
<b>4</b>	<b>System description</b>	<b>19</b>
4.1	System (in general) .....	20
4.1.1	System context of the controller .....	20
4.1.2	Standard PLC and safety PLC .....	21
4.2	Safety architecture .....	22
4.2.1	System architecture .....	23
4.2.2	Usage in applications according to ISO 13849-1 .....	25
4.2.3	Safe state .....	26
4.2.4	Diagnostics .....	27
4.3	Time response .....	29
4.3.1	The process safety time .....	30
4.3.2	Diagnostic Test Intervall (DTI) .....	32
4.3.3	Time-related behaviour of the inputs .....	32
4.3.4	Time-related behaviour IEC application .....	33
4.3.5	Time-related behaviour of the outputs .....	34
4.3.6	Safety time .....	36
4.3.7	Response time .....	37
4.3.8	Configuration time .....	37
4.4	Hardware description .....	38
4.4.1	Hardware structure .....	39
4.4.2	Device supply (technology) .....	42
4.4.3	Inputs (technology) .....	47
4.4.4	Outputs (technology) .....	52
4.4.5	Feedback in case of externally supplied outputs .....	57
4.4.6	Influence of actuators on the output diagnostics .....	57
4.5	Interfaces .....	59
4.5.1	Serial interface .....	59
4.5.2	Ethernet interface .....	59
4.5.3	CAN: Interfaces and protocols .....	60
4.6	Software description .....	61
4.6.1	Overview: Software .....	61
4.6.2	Software module for the device .....	62

<b>5</b>	<b>Installation</b>	<b>65</b>
5.1	System requirements .....	65
5.1.1	Hardware .....	65
5.1.2	Software .....	65
5.2	Carry out installation .....	66
5.2.1	CODESYS Development System .....	66
5.2.2	Complete package for ecomatController CR7xxS .....	66
<b>6</b>	<b>Getting started</b>	<b>69</b>
6.1	Use the CODESYS operating instructions .....	69
6.2	Start CODESYS .....	69
6.3	Create CODESYS project .....	70
6.3.1	Create new project with CR7xxS .....	71
6.3.2	Overview: Project structure with CR7xxS .....	72
6.4	Configure the programming interface .....	73
6.4.1	Set communication path of PLC .....	73
6.4.2	Check the communication path (blinking test) .....	74
6.5	Activate the access protection for a project .....	74
<b>7</b>	<b>System configuration</b>	<b>75</b>
7.1	Preparation of the PLC .....	75
7.1.1	Managing device users .....	76
7.1.2	Create a user in the CODESYS project .....	77
7.1.3	Sign the user in to the CODESYS project .....	78
7.1.4	Enter information about applications .....	78
7.1.5	Assign memory allocation -- safety / standard PLC .....	78
7.1.6	Assign inputs/outputs - safety / standard PLC .....	79
7.1.7	Manage files .....	81
7.1.8	User-defined data .....	83
7.2	Configuring the standard PLC .....	85
7.2.1	Configure task processing .....	86
7.2.2	Add function libraries to the application .....	87
7.3	Configuring the safety PLC .....	88
7.3.1	Configuration of the safety task processing .....	89
7.3.2	Add function libraries to the safety application .....	89
7.4	Use memory .....	90
7.4.1	Memory protection .....	90
7.4.2	Data exchange between standard PLC and safety PLC .....	92
7.4.3	Stack .....	92
7.5	Configure IEC watchdog .....	93
7.6	Configure interfaces .....	94
7.6.1	Configure serial interface .....	94
7.6.2	Configure Ethernet interface .....	94
7.6.3	Configuring CAN interfaces .....	96
7.6.4	Interface configuration file comconf.cfg .....	104
7.7	Configure inputs and outputs .....	105
7.7.1	via function block .....	105
7.7.2	via safety function block .....	106
7.7.3	via system configuration .....	106
<b>8</b>	<b>Programming of the standard application</b>	<b>107</b>
8.1	Error in the IEC application .....	107
8.2	Objects of a standard PLC application .....	108
8.3	Creating a standard PLC application .....	109
8.3.1	Supported programming languages .....	109
8.3.2	Data types .....	109

8.3.3	Supported variable types.....	110
8.3.4	Options to access the input data and output data .....	112
8.3.5	Default behaviour of the inputs and outputs in case of an error .....	113
8.3.6	Standard diagnostic limit values of inputs.....	113
8.3.7	Standard diagnostic limit values of outputs .....	114
8.3.8	Call sequence for diagnostic function blocks for inputs and outputs .....	114
8.4	Using ifm function libraries .....	116
8.4.1	Access to inputs .....	116
8.4.2	Access to outputs .....	117
8.4.3	Control device .....	117
8.4.4	Read device information.....	117
8.5	Use IO mapping .....	118
8.5.1	Access inputs .....	118
8.5.2	Access outputs .....	119
8.5.3	Accessing the system inputs .....	119
8.5.4	Accessing the system outputs .....	120
8.5.5	Accessing user LEDs .....	121
8.6	Use RawCAN (CAN Layer 2) .....	122
8.6.1	RawCAN: Control CAN network nodes .....	122
8.6.2	RawCAN: Send and receive CAN messages.....	122
8.6.3	RawCAN: Request and send remote CAN messages.....	122
8.7	Use CANopen .....	123
8.7.1	CANopen: Send and receive SDO .....	123
8.7.2	CANopen: Network Management (NMT).....	123
8.8	Use SAE J1939.....	124
8.9	Using Ethernet .....	125
8.9.1	Modbus.....	126
<b>9</b>	<b>Programming of the safety application</b>	<b>127</b>
9.1	Objects in a safety PLC application .....	127
9.2	Using libraries that are and that are not from ifm.....	128
9.3	Error in the IEC application .....	128
9.4	Creating a safety PLC application.....	129
9.4.1	Safety requirements .....	130
9.4.2	Supported programming languages .....	130
9.4.3	Complexity level / user level.....	131
9.4.4	Supported variable types.....	132
9.4.5	Access to safety input data and safety output data .....	136
9.4.6	Default behaviour of the inputs and outputs in case of an error .....	137
9.5	Safety programming.....	138
9.5.1	Safety-related applications .....	138
9.6	1-channel safety concept for inputs .....	140
9.6.1	Operation as fail-safe digital input .....	140
9.6.2	Operation as fail-safe analogue input.....	143
9.6.3	Operation as fail-safe digital input with blanking pulses .....	146
9.7	2-channel safety concept for inputs .....	150
9.7.1	fail-safe 2-channel evaluation of the inputs .....	151
9.7.2	Operation as 2-channel fail-safe digital input.....	153
9.7.3	Operation as 2-channel fail-safe analogue input .....	156
9.7.4	Operation as 2-channel fail-safe frequency input .....	160
9.8	Safety concept of the outputs .....	163
9.8.1	Operation as fail-safe digital output .....	164
9.8.2	Operation as PWM output and current-controlled output .....	166
9.8.3	Output, 1-channel, safe .....	168
9.8.4	Output 2-channel, safe, with output group.....	169
9.8.5	Output, 2-channel, safe, without output group.....	170
9.8.6	Switching off an output group safely.....	172
9.9	Use CANopen-Safety.....	176
9.9.1	CANopen safety error behaviour.....	176
9.9.2	Characteristic safety values for CANopen Safety.....	177



<b>10</b>	<b>Set-up and maintenance</b>	<b>180</b>
10.1	Connect the device to the network.....	180
10.2	Check the operating system version of the device .....	181
10.2.1	Check the operating system version of the device .....	181
10.2.2	Check the hardware version of the device.....	181
10.3	Update the operating system of the device .....	182
10.3.1	Update the operating system of the device with the ifm Maintenance Tool.....	182
10.3.2	Update the operating system of the device with the batch file.....	182
10.4	Transfer CODESYS project to device.....	184
10.4.1	Load the standard application to the device .....	184
10.4.2	Load the safety application to the device.....	185
10.4.3	Delete the application program on the device .....	185
10.5	Data transmission for series production .....	187
10.5.1	Data transmission with the ifm Maintenance Tool .....	188
10.5.2	Transmission of the files with CODESYS .....	188
10.5.3	Data transmission with TFTP .....	189
10.5.4	Files for series production .....	189
10.6	Documentation of the overall application.....	191
10.6.1	Document the serial number .....	191
10.6.2	Create a network diagram .....	191
10.6.3	Document the checksums .....	191
10.6.4	Read the device information.....	192
10.6.5	Display system information.....	195
10.7	CODESYS Debugging .....	196
<b>11</b>	<b>Operation</b>	<b>198</b>
11.1	Operating states.....	198
11.1.1	Overview of operating mode states .....	199
11.2	Status LEDs .....	201
11.2.1	Status LED: system ifm operating system (SYS0+SYS1) .....	201
11.2.2	Status LED: system PLC (SYS0, SYS1) .....	201
11.2.3	Status LED: System bootloader (SYS0) .....	202
11.2.4	Status LED: Sleep mode (SYS0).....	203
11.2.5	Status LED: Ethernet interfaces (ETH0, ETH1).....	203
11.2.6	Controlling LEDs in the applications.....	203
11.3	Reset.....	204
11.3.1	Reset behaviour of the system .....	204
11.3.2	Executing reset variants .....	205
11.3.3	Reset application (warm).....	205
11.3.4	Reset application (cold).....	205
11.3.5	Reset application (origin).....	205
<b>12</b>	<b>ifm function libraries</b>	<b>207</b>
12.1	General information .....	207
12.2	Using the function blocks .....	207
12.3	Device library .....	211
12.3.1	Library ifmDeviceCR07nn.library.....	211
12.4	CAN libraries .....	216
12.4.1	Library ifmRawCAN.library .....	216
12.4.2	Library ifmCANOpenManager.library .....	243
12.5	Input and output libraries .....	252
12.5.1	Library ifmConfigSwThreshold.library.....	252
12.5.2	Library ifmFastInput.library .....	256
12.5.3	Library ifmIOcommon.library .....	269
12.5.4	Library ifmIOconfigDiagProt.library.....	289
12.5.5	Library ifmOutGroup.....	298
12.5.6	Library ifmOutHBridge.....	303
12.5.7	Library ifmOutPWM .....	308

12.6	Help function libraries .....	317
12.6.1	Library ifmSysInfo.library .....	317
12.7	Safety libraries .....	321
12.7.1	Library ifmIOSafety.library .....	321
12.7.2	Library ifmPLCopenAddonSafe.library .....	374
12.7.3	Library ifmPLCopenSafe.library .....	393
<b>13</b>	<b>Troubleshooting</b> .....	<b>461</b>
13.1	Error classes .....	462
13.2	Error messages .....	462
13.3	Messages / diagnostic codes of the function blocks .....	463
<b>14</b>	<b>Appendix</b> .....	<b>464</b>
14.1	List of inputs .....	465
14.1.1	List of the inputs CR710S .....	465
14.1.2	List of the inputs CR711S .....	465
14.1.3	List of the inputs CR720S .....	466
14.1.4	List of the inputs CR721S .....	469
14.2	List of outputs .....	471
14.2.1	List of the outputs CR710S .....	471
14.2.2	List of the outputs CR711S .....	471
14.2.3	List of outputs CR720S .....	472
14.2.4	List of outputs .....	473
14.3	Filter .....	476
14.3.1	Filter times of the inputs .....	477
14.3.2	Filter times of the outputs .....	477
14.4	Using the operators .....	478
14.5	Behaviour in case of floating point operations .....	480
14.5.1	Behaviour in case of specific arguments .....	481
14.5.2	Behaviour in case of conversions .....	482
14.6	Behaviour in case of integer operations .....	483
14.7	Mapping table [H2] user manual / ifm ecomatController CR7xxS .....	483
14.8	Directory structure and file overview .....	485
14.9	Overview of user rights .....	487
14.10	Task configuration example .....	489
14.10.1	Current situation .....	490
14.10.2	Setting the task configuration .....	491
14.10.3	Assign the CAN manager to the bus task .....	493
14.10.4	Mapping of the CAN variables .....	494
14.10.5	Call CAN POU's .....	495
14.10.6	Task monitoring .....	496
14.11	ifm behaviour models for function blocks .....	497
14.11.1	General .....	497
14.11.2	Behaviour model ENABLE .....	497
14.11.3	Behaviour model EXECUTE .....	498
<b>15</b>	<b>Terms and abbreviations</b> .....	<b>499</b>
<b>16</b>	<b>Index</b> .....	<b>501</b>

# 1 About this manual

## Content

Legal and copyright information .....	7
Purpose of the document .....	7
Symbols used .....	8
Warnings used .....	8
Overview: User documentation for CR7xxS .....	9
Overview: documentation for CODESYS 3.n .....	9
CODESYS .....	10
Change history .....	11

26077

## 1.1 Legal and copyright information

33117

© All rights reserved by ifm electronic gmbh. No part of this manual may be reproduced and used without the consent of ifm electronic gmbh.

All product names, pictures, companies or other brands used on our pages are the property of the respective rights owners:

- AS-i is the property of the AS-International Association, (→ [www.as-interface.net](http://www.as-interface.net))
- CAN is the property of the CiA (CAN in Automation e.V.), Germany (→ [www.can-cia.org](http://www.can-cia.org))
- CODESYS™ is the property of the CODESYS GmbH, Germany (→ [www.codesys.com](http://www.codesys.com))
- DeviceNet™ is the property of the ODVA™ (Open DeviceNet Vendor Association), USA (→ [www.odva.org](http://www.odva.org))
- EtherNet/IP® is the property of the → ODVA™
- EtherCAT® is a registered trade mark and patented technology, licensed by Beckhoff Automation GmbH, Germany
- IO-Link® is the property of the → PROFIBUS Nutzerorganisation e.V., Germany (→ [www.io-link.com](http://www.io-link.com))
- ISOBUS is the property of the AEF – Agricultural Industry Electronics Foundation e.V., Deutschland (→ [www.aef-online.org](http://www.aef-online.org))
- Microsoft® is the property of the Microsoft Corporation, USA (→ [www.microsoft.com](http://www.microsoft.com))
- Modbus® is the property of the Schneider Electric SE, France (→ [www.schneider-electric.com](http://www.schneider-electric.com))
- PROFIBUS® is the property of the PROFIBUS Nutzerorganisation e.V., Germany (→ [www.profibus.com](http://www.profibus.com))
- PROFINET® is the property of the → PROFIBUS Nutzerorganisation e.V., Germany
- Windows® is the property of the → Microsoft Corporation, USA

## 1.2 Purpose of the document

58699

These instructions apply to the following devices:

- ecomatController CR710S firmware version V3.1.x.y and higher
- ecomatController CR711S firmware version V3.1.x.y and higher
- ecomatController CR720S firmware version V3.1.x.y and higher
- ecomatController CR721S firmware version V3.1.x.y and higher

The devices differ by the number of inputs and outputs and the output groups. → Data sheet

To improve readability, the mentioned devices are collectively referred to as CR7xxS in this programming manual.

These instructions describe the following topics:

- Configuration of the device using CODESYS
- Programming of the device-internal standard PLC and the safety PLC of the device using the CODESYS programming system
- Description of the device-specific CODESYS function libraries
- Updating the firmware of the device

## 1.3 Symbols used

58277



Important note  
Non-compliance can result in malfunction or interference



Information  
Supplementary note



Request for action



Reaction, result



"see"

**abc**

Cross-reference

123

Decimal number

0x123

Hexadecimal number

0b010

Binary number

[...]

Designation of pushbuttons, buttons or indications

## 1.4 Warnings used

58278



### WARNING

Warning of serious personal injury.  
Death or serious irreversible injuries may result.



### CAUTION

Warning of personal injury.  
Slight reversible injuries may result.



### NOTICE!

Warning of damage to property

## 1.5 Overview: User documentation for CR7xxS

58700

The documentation of the device consists of the following modules:

Document	Content / Description
Data sheet	Technical data
Operating instructions	<ul style="list-style-type: none"> <li>Instructions for installation, electrical installation and commissioning</li> <li>Technical data</li> </ul>
Programming manual	<ul style="list-style-type: none"> <li>Creation of a CODESYS project with this device</li> <li>Target system configuration with CODESYS</li> <li>Programming the device-internal PLC using CODESYS</li> <li>Description of the device-specific CODESYS function libraries</li> </ul>
Release notes	Additional information on the device, the software version and known issues

If any documents are not available, they can be requested from ifm or can be downloaded from the ifm website:

→ [www.ifm.com](http://www.ifm.com)

## 1.6 Overview: documentation for CODESYS 3.n

54178

CODESYS GmbH (→ [www.codesys.com](http://www.codesys.com)) provides the following user documentations to program the device with CODESYS:

Document	Content / Description
Online help	<ul style="list-style-type: none"> <li>Context-sensitive help</li> <li>Description of the CODESYS programming system</li> </ul> <p>After the installation of the programming system stored and accessible on the hard disk of the PC/laptop:          ... \Programs (x86) \3S CODESYS \CODESYS \Online Help</p> <p><b>!</b> The online help is for programming the standard PLC. The CODESYS online help only applies to the safety PLC with restrictions.</p> <p>► Please observe this programming manual when programming the safety PLC.</p>
CODESYS installation and first steps	<ul style="list-style-type: none"> <li>Remarks about the installation of the programming system CODESYS</li> <li>First steps for handling the CODESYS programming system</li> </ul> <p>After the installation of the programming system stored and accessible on the hard disk of the PC/laptop:          ... \Programs (x86) \3S CODESYS \CODESYS \ Documentation</p>
CODESYS user manual Safety SIL2	<p>[H2] CODESYS Safety SIL2 - IEC Programming Guidelines.pdf          This document is intended for programmers who program safety-related controllers.</p> <p><b>!</b> The CODESYS user manual Safety SIL2 only applies to the safety PLC with restrictions.</p> <p>► Please observe this programming manual when programming the safety PLC.</p>

If any documents are not available, they can be requested from ifm or can be downloaded from the ifm website:

→ [www.ifm.com](http://www.ifm.com)

## 1.7 CODESYS

58701

- ▶ To program the standard PLC, use version V3.5 SP11 or higher of the CODESYS programming system.
- ▶ To program the safety PLC, only use version V3.5 SP11 of the CODESYS programming system without patches.

## 1.8 Change history

60498

Date	State	Change
08 / 2020	80295862 / 00	<p>The most important changes in connection with firmware V3.1:</p> <ul style="list-style-type: none"> <li>▪ New function block descriptions: SF_Equivalent_DINT, SF_OutputEnh, SF_OutGroupEnh, SF_PWM1000Enh, SF_CurrentControlEnh, SF_HBridgeEnh, CAN_RxRangeExt</li> <li>▪ Description of new input mode IN_PERIOD_RATIO_US_CSI /CSO</li> <li>▪ Description of FB CAN_RxRange, FB CAN_Rx, FB CAN_Mask, FB CAN_Status</li> <li>▪ Description new operating state SLEEP</li> <li>▪ Description of how to use the operators, SIN, COS, TAN and restrictions</li> <li>▪ Description of the behaviour of floating point and integer operations</li> <li>▪ Description of memory protection - IO-mapping on the safety PLC</li> <li>▪ Change of company name 3S-Smart Software Solutions GmbH -&gt; CODESYS GmbH</li> </ul> <p>Amendment/Revision of the following chapters:</p> <ul style="list-style-type: none"> <li>▪ <b>Libraries</b> (→ p. <a href="#">64</a>)</li> <li>▪ <b>Behaviour in case of voltage dip</b> (→ p. <a href="#">43</a>)</li> <li>▪ <b>Sleep mode</b> (→ p. <a href="#">45</a>)</li> <li>▪ <b>Input type IN FREQUENCY-B</b> (→ p. <a href="#">49</a>)</li> <li>▪ <b>Ethernet interface</b> (→ p. <a href="#">59</a>)</li> <li>▪ <b>User-defined data</b> (→ p. <a href="#">83</a>)</li> <li>▪ <b>File system - read/write performance</b> (→ p. <a href="#">84</a>)</li> <li>▪ <b>Use memory</b> (→ p. <a href="#">90</a>)</li> <li>▪ <b>Memory protection</b> (→ p. <a href="#">90</a>)</li> <li>▪ <b>Stack</b> (→ p. <a href="#">92</a>)</li> <li>▪ <b>Non-volatile data</b> (→ p. <a href="#">110</a>)</li> <li>▪ <b>Function description of non-volatile data</b> (→ p. <a href="#">111</a>)</li> <li>▪ <b>Options to access the input data and output data</b> (→ p. <a href="#">112</a>)</li> <li>▪ <b>Standard diagnostic limit values of inputs</b> (→ p. <a href="#">113</a>)</li> <li>▪ <b>Standard diagnostic limit values of outputs</b> (→ p. <a href="#">114</a>)</li> <li>▪ <b>Using Ethernet</b> (→ p. <a href="#">125</a>)</li> <li>▪ <b>Modbus</b> (→ p. <a href="#">126</a>)</li> <li>▪ <b>Using ifm function libraries</b> (→ p. <a href="#">116</a>)</li> <li>▪ <b>Function description of non-volatile data</b> (→ p. <a href="#">111</a>)</li> <li>▪ <b>Function description of non-volatile data</b> (→ p. <a href="#">111</a>)</li> <li>▪ <b>Non-volatile safety-related data (dynamic)</b> (→ p. <a href="#">136</a>)</li> <li>▪ <b>Standard diagnostic limit values of inputs</b> (→ p. <a href="#">113</a>)</li> <li>▪ <b>Standard diagnostic limit values of outputs</b> (→ p. <a href="#">114</a>)</li> <li>▪ <b>Supported variable types</b> (→ p. <a href="#">110</a>)</li> <li>▪ <b>Supported variable types</b> (→ p. <a href="#">132</a>)</li> <li>▪ <b>Safety-related applications</b> (→ p. <a href="#">138</a>)</li> <li>▪ <b>Detailed error evaluation</b> (→ p. <a href="#">155</a>)</li> <li>▪ <b>Operation as fail-safe digital input</b> (→ p. <a href="#">140</a>)</li> <li>▪ <b>Operation as fail-safe analogue input</b> (→ p. <a href="#">143</a>)</li> <li>▪ <b>Operation as 2-channel fail-safe digital input</b> (→ p. <a href="#">153</a>)</li> <li>▪ <b>Operation as 2-channel fail-safe frequency input</b> (→ p. <a href="#">160</a>)</li> <li>▪ <b>Safety concept of the outputs</b> (→ p. <a href="#">163</a>)</li> <li>▪ <b>Operation as fail-safe digital output</b> (→ p. <a href="#">164</a>)</li> <li>▪ <b>Operation as PWM output and current-controlled output</b> (→ p. <a href="#">166</a>)</li> <li>▪ <b>Output, 1-channel, safe</b> (→ p. <a href="#">168</a>)</li> <li>▪ <b>Output 2-channel, safe, with output group</b> (→ p. <a href="#">169</a>)</li> <li>▪ <b>Output, 2-channel, safe, without output group</b> (→ p. <a href="#">170</a>)</li> <li>▪ <b>Switching off an output group safely</b> (→ p. <a href="#">172</a>)</li> </ul>

Date	State	Change
		<ul style="list-style-type: none"> <li>▪ <b>Use CANopen-Safety</b> (→ p. <a href="#">176</a>)</li> <li>▪ <b>CANopen safety error behaviour</b> (→ p. <a href="#">176</a>)</li> <li>▪ <b>CODESYS Debugging</b> (→ p. <a href="#">196</a>)</li> <li>▪ <b>Operating states</b> (→ p. <a href="#">198</a>)</li> <li>▪ <b>Overview of operating mode states</b> (→ p. <a href="#">199</a>)</li> <li>▪ <b>Status LED: Sleep mode (SYS0)</b> (→ p. <a href="#">203</a>)</li> <li>▪ <b>Reset behaviour of the system</b> (→ p. <a href="#">204</a>)</li> <li>▪ <b>Executing reset variants</b> (→ p. <a href="#">205</a>)</li> <li>▪ <b>RESET_TYPE (ENUM)</b> (→ p. <a href="#">215</a>)</li> <li>▪ <b>CAN_Rx</b> (→ p. <a href="#">225</a>)</li> <li>▪ <b>CAN_RxRange</b> (→ p. <a href="#">231</a>)</li> <li>▪ <b>CAN_RxRange</b> (→ p. <a href="#">231</a>)</li> <li>▪ <b>CAN_RxRangeExt</b> (→ p. <a href="#">234</a>)</li> <li>▪ <b>CAN_Status</b> (→ p. <a href="#">237</a>)</li> <li>▪ <b>MODE_PERIOD (ENUM)</b> (→ p. <a href="#">268</a>)</li> <li>▪ <b>CurrentControl</b> (→ p. <a href="#">309</a>)</li> <li>▪ <b>SF_InputBlanking</b> (→ p. <a href="#">327</a>)</li> <li>▪ <b>SF_Output</b> (→ p. <a href="#">357</a>)</li> <li>▪ <b>SF_OutputGroup</b> (→ p. <a href="#">360</a>)</li> <li>▪ <b>SF_CurrentControl</b> (→ p. <a href="#">364</a>)</li> <li>▪ <b>SF_PWM1000</b> (→ p. <a href="#">367</a>)</li> <li>▪ <b>SF_HBridge</b> (→ p. <a href="#">371</a>)</li> <li>▪ <b>SF_OutputEnh</b> (→ p. <a href="#">332</a>)</li> <li>▪ <b>SF_OutGroupEnh</b> (→ p. <a href="#">336</a>)</li> <li>▪ <b>SF_PWM1000Enh</b> (→ p. <a href="#">341</a>)</li> <li>▪ <b>SF_CurrentControlEnh</b> (→ p. <a href="#">346</a>)</li> <li>▪ <b>SF_HBridgeEnh</b> (→ p. <a href="#">351</a>)</li> <li>▪ <b>State diagram SF_[Type]Enh</b> (→ p. <a href="#">356</a>)</li> <li>▪ <b>Library ifmPLCopenAddonSafe.library</b> (→ p. <a href="#">374</a>)</li> <li>▪ <b>SF_Equivalent_BOOL</b> (→ p. <a href="#">375</a>)</li> <li>▪ <b>SF_Equivalent_DINT</b> (→ p. <a href="#">379</a>)</li> <li>▪ <b>SF_Equivalent_REAL</b> (→ p. <a href="#">388</a>)</li> <li>▪ <b>SF_Equivalent_UDINT</b> (→ p. <a href="#">385</a>)</li> <li>▪ <b>SF_Equivalent_UINT</b> (→ p. <a href="#">382</a>)</li> <li>▪ <b>Using the function blocks</b> (→ p. <a href="#">207</a>)</li> <li>▪ <b>Using the operators</b> (→ p. <a href="#">478</a>)</li> <li>▪ <b>Behaviour in case of floating point operations</b> (→ p. <a href="#">480</a>)</li> <li>▪ <b>Behaviour in case of specific arguments</b> (→ p. <a href="#">481</a>)</li> <li>▪ <b>Behaviour in case of conversions</b> (→ p. <a href="#">482</a>)</li> <li>▪ <b>Behaviour in case of integer operations</b> (→ p. <a href="#">483</a>)</li> <li>▪ Glossary</li> </ul>
02 / 2020	80292363 / 00	<p>Merging of the CR710S, CR711S, CR720S and CR721S programming manuals.</p> <p>Amendment/Revision of the following chapters, among others:</p> <ul style="list-style-type: none"> <li>▪ <b>Purpose of the document</b> (→ p. <a href="#">7</a>)</li> <li>▪ <b>Symbols used</b> (→ p. <a href="#">8</a>)</li> <li>▪ <b>Warnings used</b> (→ p. <a href="#">8</a>)</li> <li>▪ <b>Overview: User documentation for CR7xxS</b> (→ p. <a href="#">9</a>)</li> <li>▪ <b>CODESYS</b> (→ p. <a href="#">10</a>)</li> <li>▪ <b>Functions and features</b> (→ p. <a href="#">18</a>)</li> <li>▪ <b>Diagnostics</b> (→ p. <a href="#">27</a>)</li> </ul>



Date	State	Change
		<ul style="list-style-type: none"> <li>▪ <b>Block diagram of the supply and of the output deactivation</b> (→ p. <a href="#">45</a>)</li> <li>▪ <b>Switch off outputs via solid-state switch</b> (→ p. <a href="#">45</a>)</li> <li>▪ <b>Input type IN MULTIFUNCTION-A</b> (→ p. <a href="#">48</a>)</li> <li>▪ Input type IN FREQUENCY-B</li> <li>▪ <b>Influence of actuators on the output diagnostics</b> (→ p. <a href="#">57</a>)</li> <li>▪ <b>Assign inputs/outputs - safety / standard PLC</b> (→ p. <a href="#">79</a>)</li> <li>▪ <b>Supported programming languages</b> (→ p. <a href="#">109</a>)</li> <li>▪ <b>Supported programming languages</b> (→ p. <a href="#">130</a>)</li> <li>▪ <b>Safety data types</b> (→ p. <a href="#">133</a>)</li> <li>▪ <b>Access to safety input data and safety output data</b> (→ p. <a href="#">136</a>)</li> <li>▪ <b>Call sequence for diagnostic function blocks for inputs and outputs</b> (→ p. <a href="#">114</a>)</li> <li>▪ <b>1-channel safety concept for inputs</b> (→ p. <a href="#">140</a>)</li> <li>▪ <b>2-channel safety concept for inputs</b> (→ p. <a href="#">150</a>)</li> <li>▪ <b>Safety concept of the outputs</b> (→ p. <a href="#">163</a>)</li> <li>▪ <b>Characteristic safety values for CANopen Safety</b> (→ p. <a href="#">177</a>)</li> <li>▪ <b>ifm function libraries</b> (→ p. <a href="#">207</a>)</li> <li>▪ <b>List of inputs</b> (→ p. <a href="#">465</a>)</li> <li>▪ <b>List of outputs</b> (→ p. <a href="#">471</a>)</li> <li>▪ <b>Using the operators</b> (→ p. <a href="#">478</a>)</li> </ul>

## 2 Safety instructions

### Content

Required previous knowledge .....	14
Important standards.....	14
Note! .....	15
Start-up behaviour of the controller .....	16
IT safety .....	16
Access protection .....	16

28333

### 2.1 Required previous knowledge

25065

This document is for specialists. Specialists are people who are qualified by their appropriate training and their experience to see risks and to avoid possible hazards that may be caused during operation or maintenance of the device.

To program the device, they should also be familiar with the CODESYS 3.5 software and the CODESYS Safety SIL2 extension.

Moreover, they should be familiar with the following topics:

- Requirements on safety-relevant programming
- Standards (→ **Important standards** (→ p. 14))

The document gives information concerning the correct handling of the product.

- ▶ Read this document before use to familiarise yourself with operating conditions, installation and operation.
- ▶ Keep this document during the entire duration of use of the device.
- ▶ Follow the safety instructions.

### 2.2 Important standards

54182

Among other things, the programmer of safety-related controllers should also know and observe the content of the following standards:

Standard	Title, content
IEC 61508	Standard: Functional safety of electrical/electronic/programmable electronic safety-related systems
ISO 13849	Standard: Safety of machinery, safety-related parts of control systems <ul style="list-style-type: none"> <li>• Part 1: General design principles</li> <li>Part 2: Validation</li> </ul>
IEC 62061	Standard: Machine safety – functional safety of electric, electronic and programmable machine controllers <ul style="list-style-type: none"> <li>• Specification of the functional requirements</li> <li>• Specification of the safety requirements</li> </ul>
ISO 25119	Standard: Tractors and agricultural and forestry machines – safety-related parts of controllers

## 2.3 Note!

24490  
54183

There is no warranty for any properties that is based on information, notes and examples provided in this manual. The drawings, representations and examples imply no responsibility for the system and no application-specific particularities.

- ▶ The manufacturer of the machine/equipment is responsible for ensuring the safety of the machine/equipment.
- ▶ Follow the national and international regulations of the country in which the machine/installation is to be placed on the market!



### WARNING

Non-observance of these instructions may lead to property damage or personal injury!  
ifm electronic gmbh does not assume any liability in this regard.

- ▶ The acting person must have read and understood the safety instructions and the corresponding chapters in this manual before working on and with this device.
- ▶ The acting person must be authorised to work on the machine/equipment.
- ▶ The acting person must have the qualification and training required to perform this work.
- ▶ Adhere to the technical data of the devices!  
The up-to-date data sheet is available on ifm's website.
- ▶ Observe the installation and wiring information as well as the functions and features of the devices!  
→ supplied operating instructions or on the ifm website
- ▶ Please note the corrections and notes in the release notes for the hardware, software and documentation available on the ifm website. → [www.ifm.com](http://www.ifm.com)



### WARNING

The application programs do not function properly or the configuration is incorrect.

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function.
- ▶ The system manufacturer is responsible for the functioning of the application programs.
- ▶ For applications of functional safety, the system manufacturer must assure the conformity of the system and the corresponding application programs in accordance with the applicable regulations.
- ▶ Certification by competent organisations may be required.

## 2.4 Start-up behaviour of the controller

54186



### WARNING

Unintentional and dangerous start of machine or plant sections.

- > Risk of personal injuries and/or damage to property.
- ▶ When creating the program, the programmer must ensure that the machine or plant sections cannot start without it being intended and constitute danger after an error occurs and is eliminated (e.g. E-stop).
  - Implement restart disable.
  - In case of an error, set the outputs concerned to FALSE in the program.

A restart can, for example, be caused by:

- Voltage restoration after power failure
- Error elimination after an E-stop

To ensure reliable controller behaviour:

- ▶ Monitor the voltage supply in the application program.
- ▶ In case of an error switch off all relevant outputs in the application program.
- ▶ Monitor actuators that may cause hazardous movements additionally in the application program (feedback).
- ▶ Actuator start-up only after additional approval, e.g. via key-operated switch.
- ▶ Use function blocks from the **Library ifmPLCopenSafe.library** (→ p. [393](#)), e.g. **SF\_EmergencyStop** (→ p. [408](#)), **SF\_EnableSwitch** (→ p. [412](#)), etc.

## 2.5 IT safety

54187

### NOTICE!

If the device is operated in an unprotected network environment.

- > Unauthorised data access (read or write) is possible.
- > Unauthorised manipulation of the device functions is possible.
- ▶ Check and restrict access options to the device:
  - Restrict access to authorised persons.
  - Do not connect the device to open networks or the internet.
  - If access from the internet should be necessary, it is mandatory to choose a secure method to connect the device (e.g. VPN).

## 2.6 Access protection

25233

The access protection of the safety controller is implemented by means of the following mechanisms:

- **User administration in the project**

Each part of the safety application of a CODESYS project can be protected against non authorised access by using corresponding settings in the CODESYS user administration.

→ **Create a user in the CODESYS project** (→ p. [77](#))



To ensure the access protection of the safety application, the user must create a corresponding CODESYS user administration for each project.

- **Password of the controller**

→ **Managing device users** (→ p. [76](#))

- **Identification of the safety controller, prevent login on the wrong controller**

For successful login to the correct safety controller, at least one of the following measures must be taken depending on the safety controller:

- (Manual) verification of the safety controller's serial number → **Document the serial number** (→ p. [191](#))
- Blinking test (winking function) → **Check the communication path (blinking test)** (→ p. [74](#))
- Use different passwords for different controllers
- Operate controllers in different networks or sub-networks

### 3 Functions and features

58703

This device is used to control processes in applications. In addition, the device contains two programmable logic controllers (PLCs) that can be programmed independently with CODESYS

- Standard PLC
- Safety PLC

The device supports the implementation of safety functions up to SIL2/PLd with the safety PLC in combination with CODESYS, safety function blocks and PLCopen Safety libraries. For this purpose, the device has inputs and outputs for a 1-channel or a 2-channel safety architecture.



Information for the implementation of safety functions with the device:

- ▶ Observe the documentation provided by ifm electronic.
- ▶ Observe the specifications for IT security.
- ▶ Observe the framework conditions specified by ifm electronic.
- ▶ Observe the settings provided and documented by ifm electronic.
- ▶ Observe the information about standards → chapter **System architecture** (→ p. [23](#)) and → chapter **Important standards** (→ p. [14](#)).
- ▶ Only program the safety functions with the safety function blocks in CODESYS that are provided by ifm.
- ▶ Only execute the safety function on the safety PLC.
- ▶ Only use the most recent firmware version from ifm electronic for safety operation of the unit.
- ▶ Only use the device as safety controller when using the framework conditions specified and the settings provided and documented by ifm electronic.
- ▶ Only use the device within the limits of the technical data (→ Data sheet).
- ▶ Use of the controller in applications according to ISO 26262 is possible in accordance with ISO 26262-8, chapter 16 "Integration of safety-related systems not developed according to ISO 26262".

58703



The attainable characteristic safety values of the automation system implemented with the device depend on:

- Safety classification of the installed sensors and actuators on the local I/O interface and the CAN bus
- Wiring and arrangement of the installed sensors and actuators
- Safety-compatible programming according to the applicable standards
- Configuration of the controller

For full evaluation of the functional safety of a safety function, all requirements of the standards to be considered must be applied to the entire safety function.

# 4      System description

<b>Content</b>	
System (in general) .....	20
Safety architecture.....	22
Time response .....	29
Hardware description.....	38
Interfaces .....	59
Software description .....	61

28392

## 4.1 System (in general)

### Content

System context of the controller .....	20
Standard PLC and safety PLC .....	21

25081

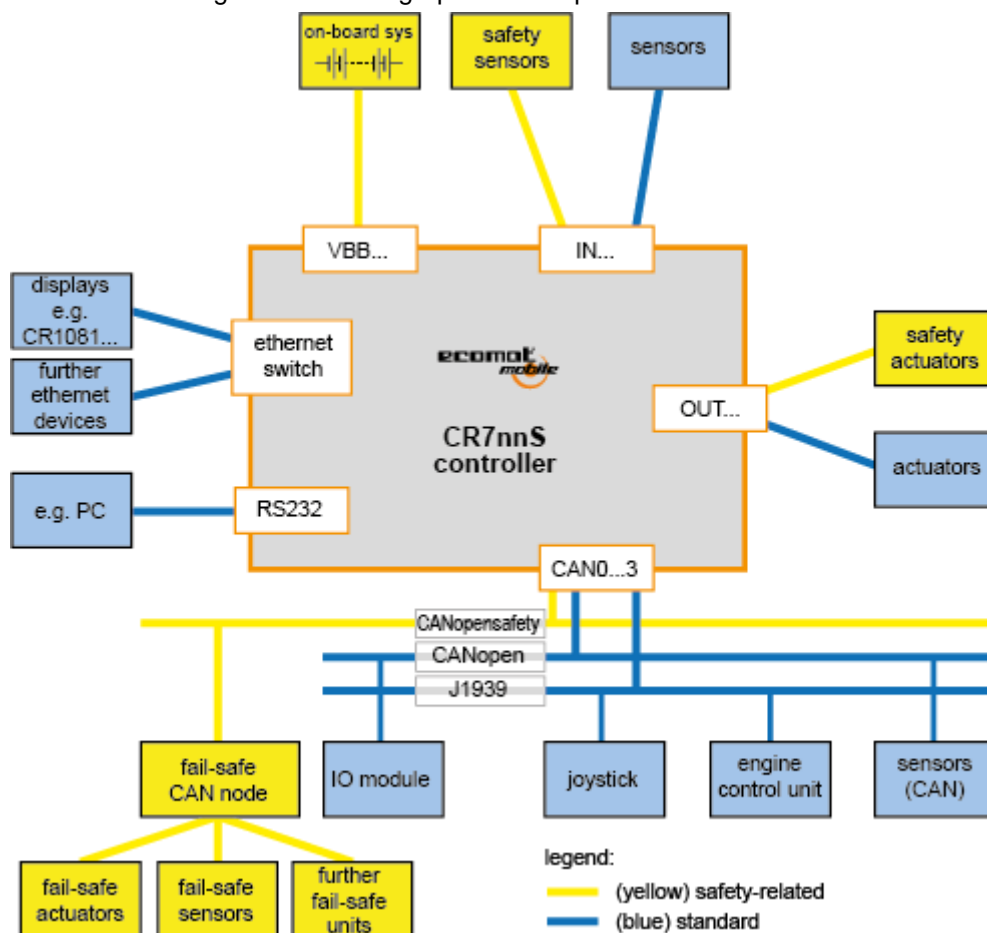
### 4.1.1 System context of the controller

24508

All devices of this controller family can be used as follows:

- PLC for safety-related application (in the figure: yellow areas)
- PLC for standard applications (in the figure: blue areas)

The elements with an orange frame in the graphics are a part of the controller.



Graphics: System context of the controller



## 4.1.2 Standard PLC and safety PLC

24495

The device features 2 separate controllers:

- for standard functions (standard PLC)
- for safety-related functions (safety PLC)

The safety and standard PLCs are free from mutual interference ("freedom of interference": memory, time-dependent behaviour and inputs and outputs).

Data can be exchanged between standard and safety PLC. (→ **Data exchange between standard PLC and safety PLC** (→ p. [92](#)))

The standard and the safety PLC are configured and programmed separately in CODESYS V3.5 SP11.

**!** Before the programming of the application may even begin:

- distribute the resources to both PLCs (→ **Preparation of the PLC** (→ p. [75](#))).  
(memory, inputs, outputs, used LEDs)

# 4.2     Safety architecture

Content	
System architecture .....	23
Usage in applications according to ISO 13849-1 .....	25
Safe state.....	26
Diagnostics .....	27

24762

## 4.2.1 System architecture

24744

The hardware structure of the CR7xxS corresponds with an implementation according to IEC 61131-6, IEC 61508 and ISO 25119 with a hardware fault tolerance (= HFT) of "0" ([ HFT = 0).

Safety characteristics and standards → Data sheet

25240

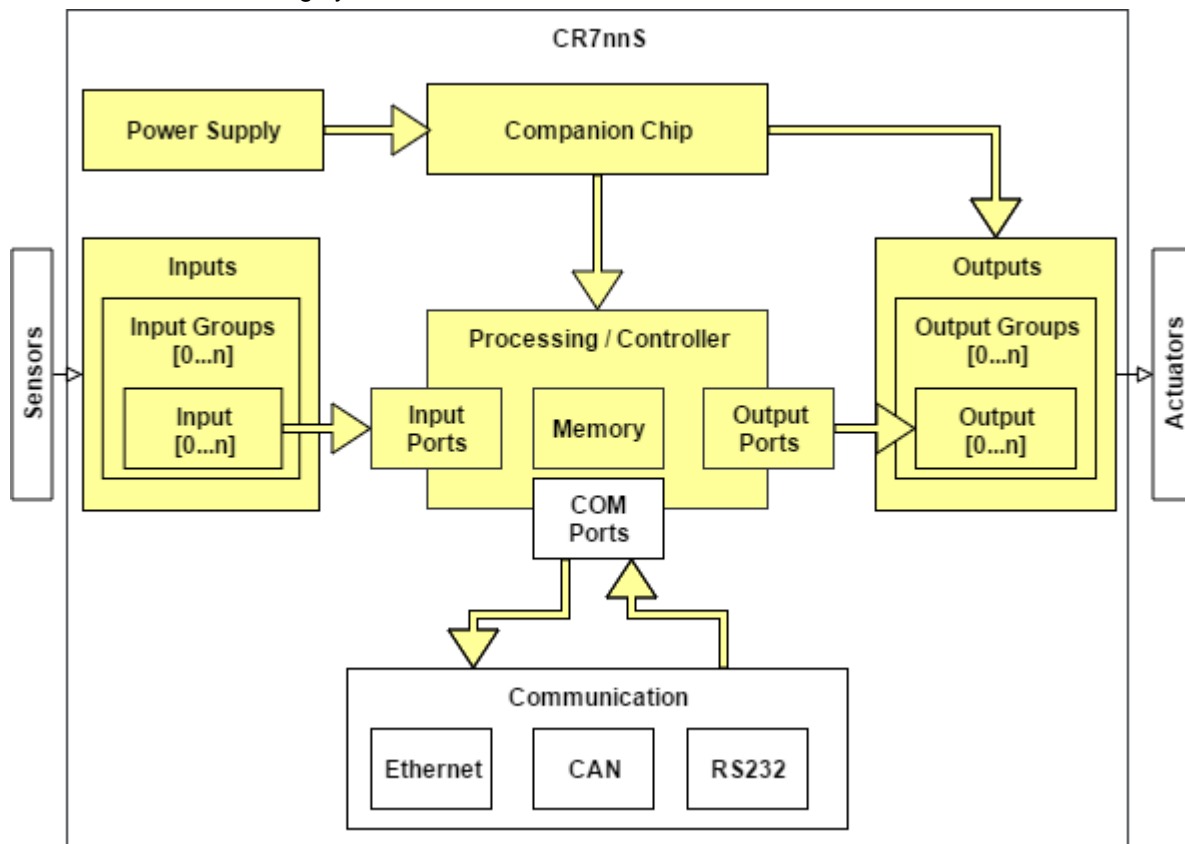


The attainable characteristic safety values of the automation system implemented with CR7xxS depend on:

- Safety classification of the installed sensors and actuators on the local I/O interface and the CAN bus
- Wiring and arrangement of the installed sensors and actuators
- Safety-compatible programming according to the applicable standards
- Configuration of the controller

For full evaluation of the functional safety of a safety function, all requirements of the standards to be considered must be applied to the entire safety function.

CR7xxS has the following system architecture:



This is implemented by means of a device architecture with the following structure features:

- modified 1oo1-HW architecture (1 out of 1 architecture) with separate test equipment
- central logic with a lock-step CPU
- Selectable:
  - 2-channel fail-safe inputs/outputs
  - 1-channel fail-safe inputs (depending on the input type)

- 1-channel fail-safe outputs
- Companion chip to monitor the main CPU and the voltage supplies with central deactivation
- A layered error concept is available (→ **Error classes** (→ p. [462](#))):
  - Errors due to which the integrity of the entire controller is no longer guaranteed will lead to the safe state of the entire controller
  - Errors due to which the integrity of a PLC is no longer guaranteed will lead to the safe state of the corresponding PLC and the assigned inputs/outputs
  - Errors of a component due to which the component can no longer be operated will lead to deactivation of the entire component or group
  - Errors of a periphery due to which an individual input/output can no longer be operated will lead to deactivation of the input/output

25137



## WARNING

Risk of incorrect configuration

Different deactivation behaviour in relation to the error class and the set diagnostic behaviour and protective behaviour (DiagMode):

- Factory setting results in the deactivation behaviour described in the chapter on error classes. → **Error classes** (→ p. [462](#))
- If settings are changed, the following behaviour is additionally to be taken into consideration: → **eDIAG\_PROT\_MODE (ENUM)** (→ p. [297](#))
  - > Risk of personal injuries and/or damage to property.
  - > Failure of the safety function is possible.
  - > In case of a fault, the machine / plant will not go into the safe state.
- ▶ Ensure fail-safe diagnostics and protective behaviour of the input and output channels by taking appropriate measures.

## 4.2.2 Usage in applications according to ISO 13849-1

The controller supports the following periphery connections for use in applications according to ISO 13849-1:

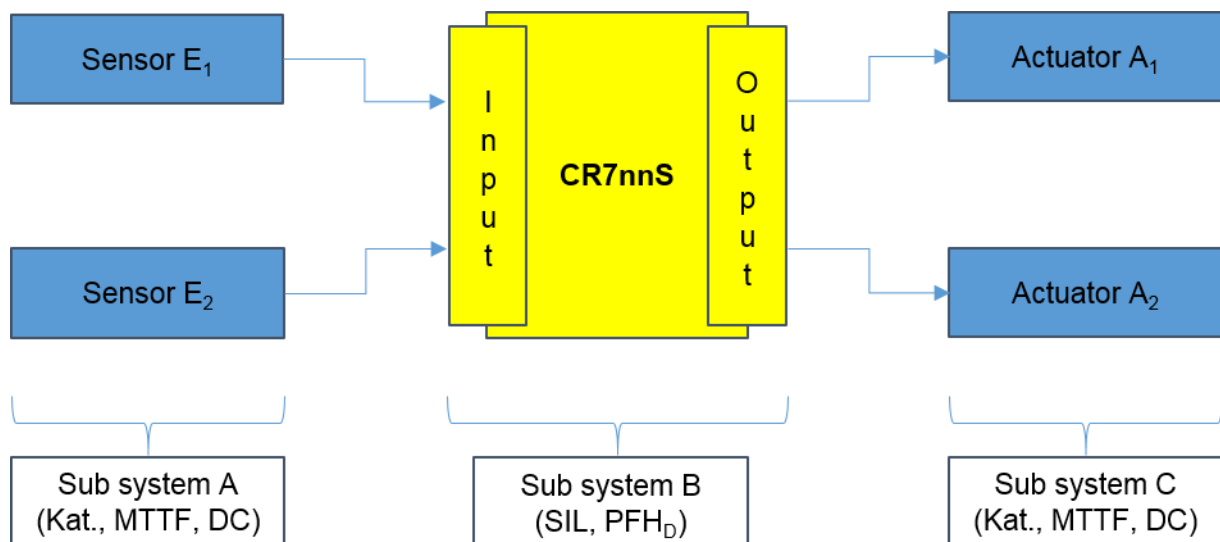
- **1-channel safety concept for inputs** (→ p. [140](#))
- **2-channel safety concept for inputs** (→ p. [150](#))
- **Safety concept of the outputs** (→ p. [163](#))

For all above-mentioned connections, the controller guarantees compliance with IEC 62061 / IEC 61508 concerning SIL and corresponding PFH<sub>D</sub>. Therefore, the controller can be used with all peripheral devices irrespective of the design. Periphery connections according to category B, 1, 2, and 3 can be implemented.

### Safety characteristics of the application:

To determine the safety characteristics of the application, the following procedure is suitable (according to ISO 13849-1, ISO/TR 23849, IFA Report 2/17):

Example (2-channel sensor/actuator connection):



- ▶ The following applies according to ISO 13849-1 Chapter 6.3:
  - $PFH_{D\text{ overall}} = PFH_{D1} + PFH_{D2} + \dots + PFH_{DN}$
- ▶ For the sub-system "A" (sensors E<sub>x</sub>) and the sub-system "C" (actuators A<sub>x</sub>):
  - Determination / calculation of category, DC<sub>avg</sub> and MTTF<sub>D</sub> according to ISO 13849-1
  - Determination of the PFH<sub>D-A</sub> / PFH<sub>D-C</sub> from the table K.1 annex K ISO 13849-1
- ▶ For the sub-system "B" (controller with "input", "logic" and "output"), add the corresponding PFH<sub>D</sub> (here: "Input 2-channel" "logic" and "output 2-channel"):
  - $PFH_{D-B} = PFH_{D\text{Input}} + PFH_{D\text{Logic}} + PFH_{D\text{Output}}$
- ▶ Determine PFH<sub>D</sub> for the safety function / the overall system:

- $PFH_{D \text{ overall}} = PFH_{D-A} + PFH_{D-B} + PFH_{D-C}$

- Read the reached PL from ISO 13849-1 table 2 and compare it to the required PL<sub>r</sub>

#### **Safety characteristics of the controller:**

Valid safety characteristics, see → operating instructions

#### **Library for SISTEMA**

For the software assistant SISTEMA from IFA ( → [www.dguv.de/ifa](http://www.dguv.de/ifa)), ifm offers a library with the characteristics of the controller.

#### **If you use SISTEMA, please consider the controller as follows:**

- Create sub-systems "A", "C" (sensors and actuators)
- Create sub-system "B" (controller) as [Subsystem(e)]. Select the option [Enter SIL/PFHD directly (manufacturer ensures compliance with the requirements of the SIL acc.to IEC 62061)].

Further steps are carried out by SISTEMA.

### **4.2.3 Safe state**

24992

The controller supports applications for which the safe state in a safety function is reached by the power-free state of the corresponding outputs.

The following applies to all **outputs**:

- The safe state of the output is always the power-free status (< 2 V AND < 25 mA).
- The safe state is reached by requesting the safety function of the output.
- The safe state is reached due to an error on the safety function of the output.

The following applies to all **inputs**:

- The safety function of an input provides the value 0 (0 V, 0 mA, FALSE, etc.) as safe state at the function block.
- The safe state is reached by requesting the safety function of the input.
- The safe state is reached due to an error on the safety function of the input.

The following applies for all **CANopen safety nodes**:

- The concerned slave nodes report errors.
- Error evaluation with IEC application.
- If an error has been detected: Ensure the safe state of the dependant safety function by means of the IEC application.
- The communication of CANopen Safety stops.
- The safe state is reached because of a communication error. → **CANopen safety error behaviour** (→ p. [176](#))

## 4.2.4 Diagnostics

### Content

Cyclic diagnostics .....	27
Self-test at restart (start-up test).....	28
Undetected errors (latent errors) .....	28

58706

The controller has the following internal diagnostics:

- **Cyclic diagnostics** (→ p. [27](#))
- **Self-test at restart (start-up test)** (→ p. [28](#))

The controller provides function libraries for the IEC application for creating application-specific diagnostics (→ **Safety libraries** (→ p. [321](#)))

### Cyclic diagnostics

58707

The internal cyclic diagnostics performed by the controller at start-up and during operation include:

- diagnosis of the supply for valid voltage values
- diagnosis of the processing including monitoring of the memory
- diagnosis of the inputs (→ **1-channel safety concept for inputs** (→ p. [140](#)), → **2-channel safety concept for inputs** (→ p. [150](#)))
- diagnosis of the outputs and output groups (→ **Safety concept of the outputs** (→ p. [163](#)))

The diagnostics are designed such that errors are detected within the safety time (→ **Safety time** (→ p. [36](#))) and lead to a switch-off depending on the error classes (→ **Error classes** (→ p. [462](#))).

### WARNING

Danger due to unintentional deactivation of all outputs if monitoring routines detect a system error:

- > Risk of personal injuries and/or damage to property.
- > The device switches off the energy for all outputs of the affected output groups.
- ▶ Note:
  - chapter **Output groups** (→ p. [40](#))
  - chapter **List of outputs** (→ p. [471](#))

### Watchdog

58708

The watchdog is a cyclic diagnosis executed in the system in several stages:

- IEC task-related watchdog  
This watchdog works in the ifm operating system and is executed separately for each PLC. Each IEC task is monitored individually.  
If an error occurs, the system only deactivates the affected PLC and the corresponding outputs.  
Error class = B
- External watchdog  
If an error occurs, this watchdog puts the entire system into the "safe state" (emergency stop). The output groups and all outputs of the group go to logical "0".  
Error class = A

→ chapter **Error classes** (→ p. [462](#))

To eliminate the fault:

- Rebooting the PLC is necessary via voltage on/off.

## Self-test at restart (start-up test)

58709

After the restart, the controller performs the following internal self-tests (start-up test) once:

- entire diagnostic path of the supply
- entire diagnostic path of the watchdogs in the companion chip
- function of the outputs and output groups and their diagnostic path (→ **Safety concept of the outputs** (→ p. [163](#)))
- function of processing / controller incl. memory and peripherals and their diagnostic path

Inputs can only be partially tested during the start-up test. The signal path cannot be tested, since the connected external signals (sensors, switches, signal cables, etc.) are required for this (→ **1-channel safety concept for inputs** (→ p. [140](#)), → **2-channel safety concept for inputs** (→ p. [150](#))).

## Undetected errors (latent errors)

58710

Irrespective of the cyclic diagnostics, errors can occur during operation which are not immediately detected by the diagnostics (latent errors). The latent errors do not immediately affect the function. The error only becomes critical if the function fails and the diagnosis does not detect this failure.

Example: The diagnostic path for the overcurrent detection no longer functions. The input or output still functions. The error would only lead to a critical failure if an overcurrent occurred without the diagnostics detecting it.



- To minimise the risk of a latent error, perform the following tests on a regular basis:
  - test of the inputs: sufficiently frequent signal change (→ **1-channel safety concept for inputs** (→ p. [140](#)), → **2-channel safety concept for inputs** (→ p. [150](#)))
  - test of the outputs and output groups: sufficiently frequent signal change (→ **Safety concept of the outputs** (→ p. [163](#)))
  - start-up test through restart for processing/controller incl. memory and peripherals, watchdog (can be initiated through PowerOn or the IEC application using the Reset function block).

A suitable time interval for the execution of the tests must be derived from the safety concept for the application or the applicable product standards of the application.



## 4.3 Time response

### Content

The process safety time .....	30
Diagnostic Test Intervall (DTI) .....	32
Time-related behaviour of the inputs .....	32
Time-related behaviour IEC application .....	33
Time-related behaviour of the outputs .....	34
Safety time .....	36
Response time .....	37
Configuration time .....	37

24745



### WARNING

When changing the following presets:

- Task configuration
  - Detection time for inputs and outputs
  - Filter settings for inputs and outputs
- > Delay of the safety function.
- > Delay of the processing time.
- > Risk of personal injuries and/or damage to property.
- Recalculation of the processing times and safety time.
- Ensure that the safety time is sufficient to comply with the process safety time.

The following times are relevant for the controller and are indicated separately for the individual components of the controller:

- Processing times
- Error detection times
- Fault response times

The processing times and safety time of the controller can be calculated from these times.

All indicated times always refer to the controller up until its input and output terminals. The times of the connected sensors and actuators are not taken into consideration.

The following table shows the meaning of the individual components:


Abbreviation	Component	Description
T(Input)	Signal propagation time of the inputs (signal processing time of the inputs)	Duration between application of the signal at the input terminal and the time when it is available for processing in the IEC application.
FDT(Input)	Failure detection time of the inputs	Duration until a fault on the input is detected by the controller.
FRT(Input)	Failure response time of the inputs	Duration until a response / error message is issued after a failure has been detected at the input.
T(IEC)	Processing time of the IEC application	Duration for the processing of the IEC application.

Abbreviation	Component	Description
FDT(IEC)	Failure detection time of the IEC application	Duration until a fault of the IEC application is detected by the controller.
FRT(IEC)	Failure response time of the IEC application	Duration until a response / error message is issued after a failure has been detected in the IEC application.
T(Output)	Signal propagation time of the outputs	Duration between the signal change in the IEC application and the time when the output terminal is changed.
TM(Output)	Signal propagation time of the output measurement	Duration between the measurement of the signal at the output terminal and the time when it is available for processing in the IEC application.
FDT(Output)	Failure detection time of the outputs	Duration until a failure at the output is detected by the controller.
FRT(Output)	Failure response time of the outputs	Duration until the output is switched off after a failure has been detected.
T(OutputGroup)	Signal propagation time of the output groups	Duration between the signal change in the IEC application and the time when the output terminal is changed.
T(Filter)	Filter time	Filter time that corresponds with the set input filter / output filter.
T(Controller)	Response time of the entire controller	Sum of the processing times of the individual components.

### 4.3.1 The process safety time


24763

The process safety time is the maximum time frame that is available before a dangerous error in the system can cause damage.


**Summary**

- ▶ Determine the admissible process safety time of the safety function.
- ▶ Check whether the application can reach the safe state within the process safety time if an error occurs.
- The safety time of the safety PLC must be shorter than the process safety time of the safety function.

- ▶ When setting up the safety function, also consider the process safety time of the application! When an error has occurred in a safety function, the application must have responded and have reached the safe state within the process safety time.


The permissible process safety time must be longer than the sum of the...

- fault detection time
- failure response time in the safety PLC of the CR7xxS
- actuator switching time

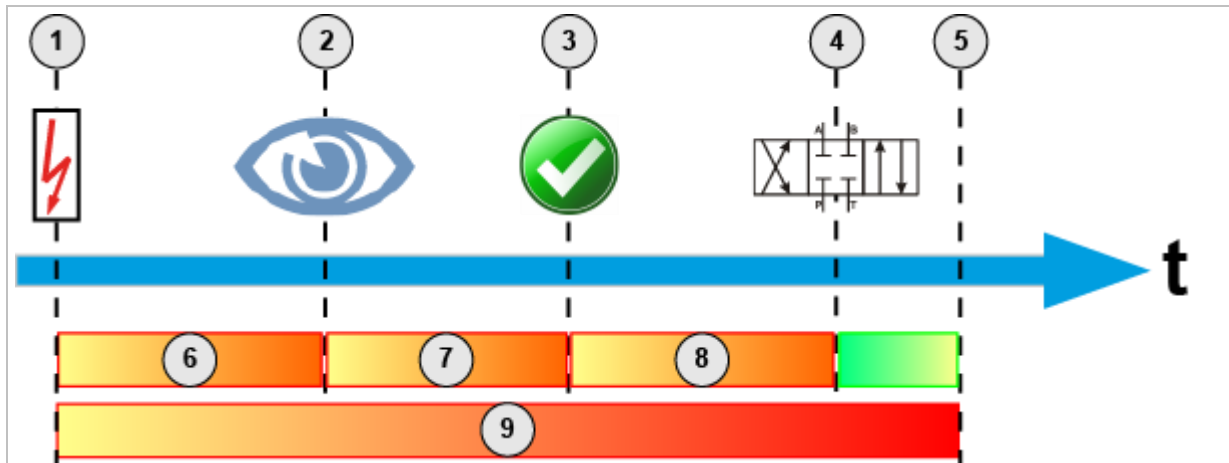


Figure: process safety time

## Legend:

- (1) Fault occurrence in the safety function
- (2) Safety PLC has detected a failure
- (3) Safety status in the safety PLC has been reached
- (4) Process in the safe state
- (5) Process safety time elapsed
- (6) Fault detection time
- (7) Failure response time in safety PLC
- (6)+(7) Safety time of the safety PLC
- (8) Switching times of the actuators
- (9) Permissible process safety time

- Take into account other potential delays caused by upstream or downstream components (sensors, actuators) for time-related considerations. These times increase the failure response time.

If data of the safety function is transferred from one safety PLC to another via the network, consider the increased failure detection and failure response time!

- For the other components in the safety function → Data sheets of the manufacturer.

For simple components (e.g. switches, valves) the following applies:

- use the diagnostic function of the controller for inputs and outputs!

>  $\text{Safety time}_{\text{total}} = \text{safety time}_{\text{safety PLC}} + \text{switching times}_{\text{actuators}}$

Safety time = failure detection time + failure reaction time

For the safety time<sub>safety PLC</sub> of the ecomatController CR7xxS, the following applies → **Safety time** (→ p. 36)

**!** In case of both single and dual channel use:

- Execute the diagnostic measures implemented in the application program in every IEC cycle to ensure that the Safety PLC safety time can be reached!

If, as described above, the safety time is less than the process safety time required by the safety function of the mobile machine, a single fault can (in the worst case) lead to a faulty output signal for a short time, but not to a loss of the safety function.

A loss of the safety function can only occur if the faulty signal cannot be corrected within the process safety time.

- Ensure that the process safety time is observed by designing the specific application in the application accordingly!

In the following cases, a single fault in the controller cannot lead to a hazardous situation:

- if the safe state (outputs OFF) is assumed...
  - after a known error or

- at the next demand upon the safety function or
- before the next demand upon the safety function.
- if the fault detection and the reaction to the fault happen within the process safety time.

### 4.3.2 Diagnostic Test Intervall (DTI)

25355

The diagnostic test interval (DTI) is defined as follows:

$$DTI = 20ms$$

### 4.3.3 Time-related behaviour of the inputs

54287

#### Signal propagation time of the inputs

##### General:

$$T(\text{Input}) = T(\text{Filter}) + \text{processing time in firmware}$$

##### Maximum, nominal and minimum:

$$T_{\text{Max}}(\text{Input}) = T_{\text{Max}}(\text{Filter}) + 2.5 \text{ ms}$$

$$T_{\text{Nom}}(\text{Input}) = T_{\text{Nom}}(\text{Filter}) + 2.0 \text{ ms}$$

$$T_{\text{Min}}(\text{Input}) = 0 \text{ ms}$$



- Set filter time → **Filter** (→ p. [476](#))
- With  $T_{\text{Nom}}(\text{Input}) / T_{\text{Min}}(\text{Input})$ : The filter can be neglected as long as the signal change is not significant. → **Filter** (→ p. [476](#))
- In case of edge signals: Depending on the mode, a signal will only be considered as complete if all criteria for signal calculation have arrived at the input. Use 0 ms as set filter time.

#### Failure detection time of the inputs

$$FDT_{\text{max}}(\text{input}) = \text{signal propagation time input with system filter}_{\text{max}} + DTI + \text{set detection time}$$

$$FDT_{\text{min}}(\text{input}) = \text{set detection time}$$

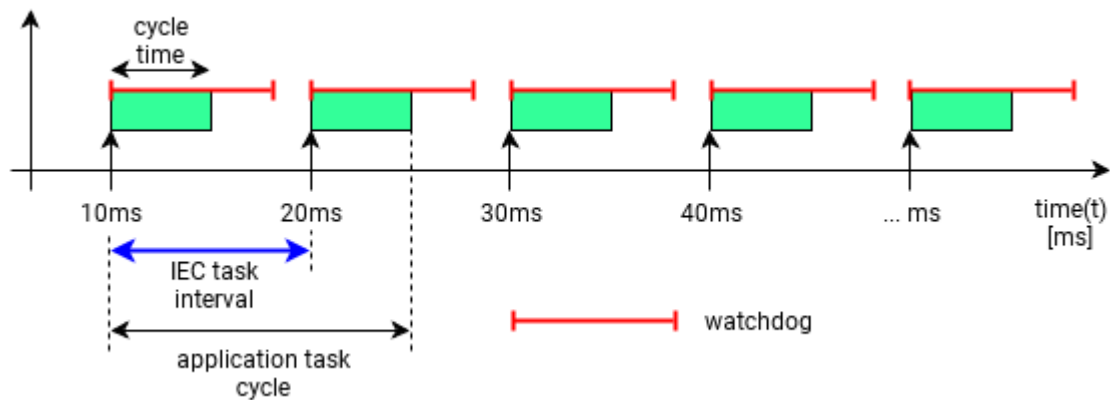


- Signal propagation of the time input with system filter<sub>max</sub> = 3.9 ms
- With  $FDT_{\text{min}}(\text{input})$ : The filter can be neglected as long as the signal change is not significant. → **Filter** (→ p. [476](#))

### 4.3.4 Time-related behaviour IEC application

54288

IEC applications have an IEC task interval in which the IEC application task is called and a cycle time (execution time of the task).



The cycle time may fluctuate during operation. The determination of the cycle time can be done by measuring the processing time, e.g. in CODESYS Task Monitoring.

The Application Task Cycle is defined as follows:

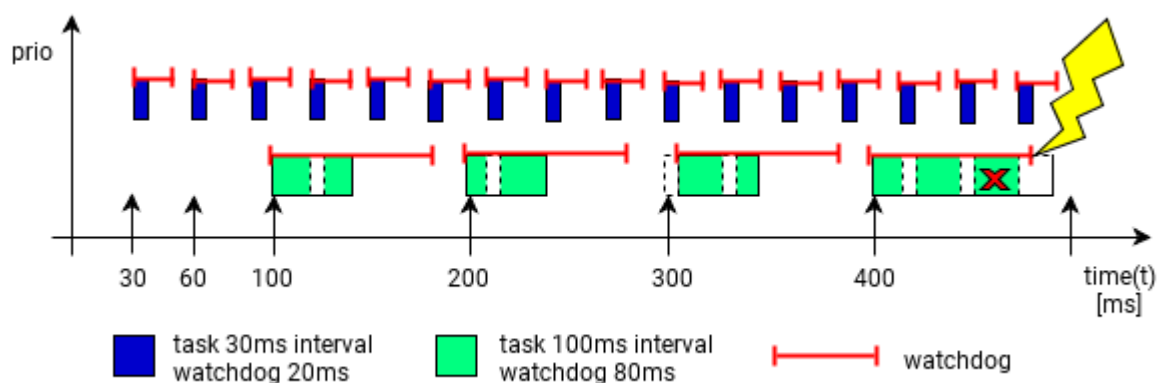
$$APPTCY = \text{IEC Task Interval} + \text{Cycle Time}_{\text{Max}}$$

In the supplied standard package, the CODESYS package features a template containing one task with standard settings per PLC and one program without function.

During the projecting time, the project engineer can adapt the template to the conditions of the overall application. (Standard application: → **Configure task processing** (→ p. 86), Safety Application: → **Configuration of the safety task processing** (→ p. 89))

Within the scope of the adaptation to the overall application, the project engineer can create more than one IEC application.

The system is pre-emptive, so that the task with higher priority will interrupt the task with lower priority.



Moreover, it is required for the overall application with safety function that the project engineer will set an IEC watchdog with an adequate time for the IEC tasks.

If the cycle time is exceeded, the watchdog will disable the application, and its resources will go into the safe state.

### Application Monitoring Time

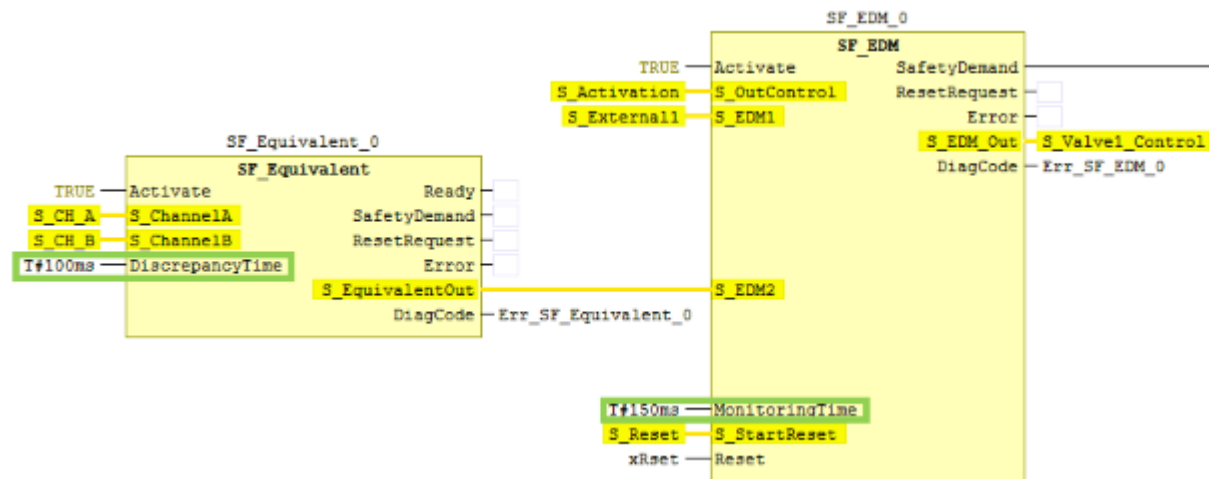
The Application Monitoring Time is a result of set diagnostic times on function blocks in the IEC application.

The following rules apply:

- If several function blocks are chained, the diagnostic times must be added up.
- If the function blocks are used in parallel, the longer diagnostic time must be used for calculation.

For example:

Application Monitoring Time = 100 ms + 150 ms = 250 ms



### Processing time of the IEC application

$T_{Max}(IEC) = 2 * IEC \text{ Task Interval}_{Max}$

$T_{Nom}(IEC) = IEC \text{ Task Interval}_{Max} + \text{Cycle Time}_{Max}$

$T_{Min}(IEC) = 0ms$

### Failure detection time of the IEC application

$FDT_{Max}(IEC) = IEC\text{-Task Interval}_{Max} + IEC\text{-Task Watchdog} + DTI / 2 + \text{Application Monitoring Time}$

$FDT_{Min}(IEC) = IEC\text{-Task Watchdog} + \text{Application Monitoring Time}$

## 4.3.5 Time-related behaviour of the outputs

54289

### Signal propagation time in general:

$T(\text{Output}/\text{OutputGroup}) = \text{processing time in firmware} + \text{processing time of the hardware}$

### Signal propagation time of the output groups

**Maximum, nominal and minimum:**

$T_{Max}(\text{OutputGroup}) = DTI / 2 + 20 \text{ ms}$

$T_{Nom}(\text{OutputGroup}) = 1 \text{ ms} + 17 \text{ ms}$

$T_{Min}(\text{OutputGroup}) = 16 \text{ ms}$



The time applies to the deactivation of an entire output group via software or in case of errors requiring a reaction of the output group.

**Signal propagation time of the outputs****Maximum, nominal and minimum:**

$$T_{\text{Max}}(\text{Output}) = \text{DTI} / 2 + 2 \text{ ms}$$

$$T_{\text{Nom}}(\text{Output}) = 1 \text{ ms} + 1 \text{ ms}$$

$$T_{\text{Min}}(\text{Output}) = 0 \text{ ms}$$



The time applies to digital outputs and the deactivation of PWM signals (value 0 / FALSE). For PWM, the value > 0 will only become active after the expired PWM period / dither period.

**Signal propagation time of the output measurement****In general, for outputs in digital operation:**

$$T_M(\text{Output}) = \text{set filter time} + \text{processing time in firmware}$$

**For outputs in digital operation****Maximum, nominal and minimum:**

$$T_{M_{\text{max}}}(\text{output}) = \text{set filter time}_{\text{max}} + 2.5 \text{ ms}$$

$$T_{M_{\text{nom}}}(\text{output}) = \text{set filter time}_{\text{nom}} + 2.0 \text{ ms}$$

$$T_{M_{\text{min}}}(\text{output}) = 0 \text{ ms}$$



- Set filter time → **Filter** (→ p. [476](#))
- For  $T(\text{output}_{\text{nom}}) / T(\text{output}_{\text{min}})$ : The filter can be neglected as long as the signal change is not significant. → **Filter** (→ p. [476](#))

**In general, for outputs in PWM operation:**

$$T_M(\text{Output}) = \text{period duration} + \text{processing time in software}$$

**Output type PWM-n-A, PWM-n-BRIDGE-A****Maximum, nominal and minimum:**

$$T_{M_{\text{Max}}}(\text{Output}) = \text{period duration} + 2.5 \text{ ms}$$

$$T_{M_{\text{Nom}}}(\text{Output}) = \text{period duration} + 2.0 \text{ ms}$$

$$T_{M_{\text{Min}}}(\text{Output}) = \text{period duration}$$



For the period duration, the following applies:

- If dither is used: Period duration = dither period duration
- otherwise: period duration = PWM period duration

**Output type PWM-n-B****Maximum, nominal and minimum:**

$$T_{M_{\text{Max}}}(\text{Output}) = 2 * \text{PWM period duration} + 2.5 \text{ ms}$$

$$T_{M_{\text{Nom}}}(\text{Output}) = 2 * \text{PWM period duration} + 2.0 \text{ ms}$$

$$T_{M_{\text{Min}}}(\text{Output}) = \text{PWM period duration}$$

**Failure detection time of outputs / output groups****General:**

$$FDT_{\text{Max}}(\text{Output}) = TM_{\text{Max}}(\text{Output}) + DTI + \text{set Detection Time}$$

$$FDT_{\text{Min}}(\text{Output}) = \text{set Detection Time}$$


- Set filter time → **Filter** (→ p. 476)
- For  $T_{\text{Nom}}(\text{Output}) / T_{\text{Min}}(\text{Output})$ : The filter can be neglected as long as the signal change is not significant. → **Filter** (→ p. 476)
- The failure response of the outputs is independent of the IEC task interval.

#### Failure response time of the outputs

Since the signal propagation time of the output groups and the outputs are different, the longer signal propagation time of the output group is considered for  $FDT_{\text{Max}}(\text{Output})$ .

$$FRT_{\text{Min}}(\text{Output}) = 0 \text{ ms}$$

$$FRT_{\text{Max}}(\text{Output}) = T_{\text{Max}}(\text{OutputGroup})$$

### 4.3.6 Safety time

24764

The safety time indicates how long it takes for the safety PLC to detect an error and to respond to the error:

$$\text{Safety time}_{\text{safety PLC}} = \text{failure detection time} + \text{failure response time}$$

The safety time of the controller is constituted by several components:

$$\text{Safety time}_{\text{safety PLC}} = FDT_{\text{max}}(\text{input} / \text{IEC application} / \text{output}) + 2 * \text{IEC task interval} + FRT_{\text{max}}(\text{output}) + DTI / 2$$


The following applies to the calculations:

- $FDT_{\text{max}}(\text{input} / \text{IEC application} / \text{output})$  is the highest value of the calculated  $FDT_{\text{max}}$  values.
- The task type must be `cyclic`
- The watchdog time must be shorter than the set IEC task interval

#### Times:

Description	Default duration
Diagnostic test interval time (DTI)	20 ms
$FDT_{\text{max}}(\text{input})$	34 ms
IEC task interval	10 ms
$FDT_{\text{max}}(\text{IEC})$	18 ms
$FDT_{\text{max}}(\text{output})$	35 ms
$FRT_{\text{max}}(\text{Output})$	30 ms

#### Example:

##### Default safety time for an IEC application task with digital outputs:

$$\text{Safety time}_{\text{safety PLC}} = 35 \text{ ms} + (2 * 10 \text{ ms}) + 30 \text{ ms} + (20 \text{ ms} / 2) = 95 \text{ ms}$$



**Safety time minimum:**

The following table shows the minimum safety time that can be reached by configuring the task interval and the filter setting and by considering the watchdog:

Safety time	Interval	Filter setting
Safety time <sub>Safety PLC min</sub> = 85 ms	4 ms	Default setting

**4.3.7 Response time**

The response time is the sum of the processing times of the individual components:

**General:**

$$T(\text{Controller}) = T(\text{Input}) + T(\text{IEC}) + T(\text{Output})$$

**Maximum, nominal, minimal:**

$$T_{\max}(\text{Controller}) = T_{\max}(\text{Input}) + T_{\max}(\text{IEC}) + T_{\max}(\text{Output})$$

$$T_{\text{nom}}(\text{Controller}) = T_{\text{nom}}(\text{Input}) + T_{\text{nom}}(\text{IEC}) + T_{\text{nom}}(\text{Output})$$

$$T_{\min}(\text{Controller}) = 0 \text{ ms}$$

**4.3.8 Configuration time**

24765

In the application, the configuration of system components can be done, e.g. mode settings. After a change in the configuration, it can take up to 20 ms until the new values are valid.

# 4.4     Hardware description

<b>Content</b>	
Hardware structure .....	39
Device supply (technology) .....	42
Inputs (technology) .....	47
Outputs (technology) .....	52
Feedback in case of externally supplied outputs .....	57
Influence of actuators on the output diagnostics .....	57

28381

## 4.4.1 Hardware structure

### Content

Overview: Hardware .....	39
Note on wiring.....	39
Available memory .....	40

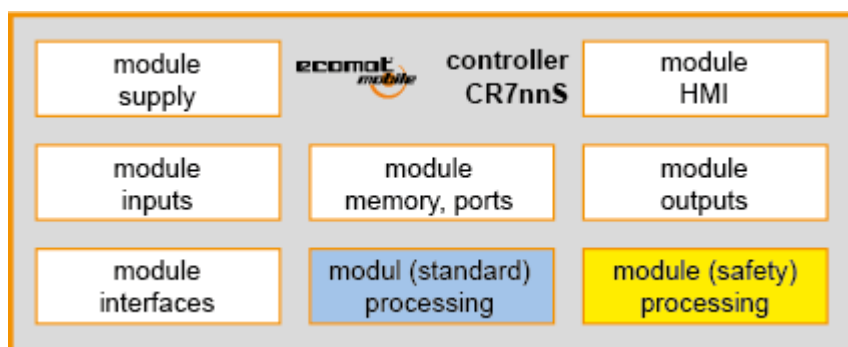
28382

### Overview: Hardware

39685

### System overview

39627



Overview of the system modules

Details:

Module	see
Power supply	<b>Device supply (technology)</b> (→ p. <a href="#">42</a> )
Inputs	<b>Inputs (technology)</b> (→ p. <a href="#">47</a> )
Interfaces	<b>Interfaces</b> (→ p. <a href="#">59</a> )
Memory, ports	<b>Available memory</b> (→ p. <a href="#">40</a> )
Processing	<b>Standard PLC and safety PLC</b>
HMI	<b>Status LEDs</b> (→ p. <a href="#">201</a> )
Outputs	<b>Outputs (technology)</b> (→ p. <a href="#">52</a> )

### Note on wiring

39414

For more detailed information → operating instructions.

### Group designations

25086

Inputs and outputs are assigned in groups.

Input groups are required for 2-channel safety-related inputs.

The identifier of an input or output results from the following principle:

1. Type (IN, OUT)
2. Group number (00...18)
3. Channel number (00...08)

Examples:

- IN0002 = input | group 00 | channel 02 in this group
- OUT0507 = output | group 05 | channel 07 in this group

## Output groups

25087

Each output group has its own supply that can be switched off (via solid-state switch).

- VBB<sub>0</sub> supplies the outputs OUT<sub>0000</sub>...OUT<sub>0008</sub>
- VBB<sub>1</sub> supplies the outputs OUT<sub>0100</sub>...OUT<sub>0108</sub>
- VBB<sub>2</sub> supplies the outputs OUT<sub>0200</sub>...OUT<sub>0208</sub>
- etc.

The concrete number of groups and the corresponding outputs is specific for each controller.

Preset:

The group switch goes on as soon as the PLC goes into RUN.

If the group switch has been turned off, the corresponding outputs will be deactivated.

## Available memory

28798

## Memory allocation

24509

The controller subdivides the memory according to IEC61131-1 to store the user data into:

- memory of the applications (parts are configurable):
  - IEC code non-safe
  - IEC code safe
- Application data memory (IEC data):
  - for volatile data (IEC RAM)
  - for non-volatile data (memory-remanent in case of voltage failure)

The device has the following additions:

- USER files (storage of application-specific data as files in the folder data, → **User-defined data** (→ p. 83))
- IEC memory bytes (permanent storage of application-specific data at application-specific addresses)

FLASH memory	RAM	non-volatile memory
physical: 9.0 Mbytes available: 5.5 Mbytes	physical: 2.7 Mbytes available: 1.5 Mbytes	physical: 10 kByte available: 10 kByte
IEC code (safe) (configurable)	IEC-RAM (safe) (configurable)	IEC-Retain (safe) (2.5 Kbyte)
IEC code (non safe) (configurable)	IEC-RAM (non safe) (configurable)	IEC memory bytes (safe) (2.5 Kbyte)
user files (1.0 Mbytes)		IEC retain (non-safe) (2.5 Kbyte)
		IEC memory bytes (non-safe) (2.5 Kbyte)

Table: Memory areas of the controller that have been made available

## Memory allocation variants

54214

The user can select from the pre-defined configurations of the memory layout. The configuration enables optimum separation between safety-relevant application and standard application.

Memory Configuration	IEC code Safety PLC	IEC-RAM Safety PLC	IEC code Standard PLC	IEC-RAM Standard PLC
Configuration 1	1.0 Mbytes	512 kByte	3.5 Mbytes	1024 kByte
Configuration 2 (preset)	2.25 Mbytes	768 kByte	2.25 Mbytes	768 kByte
Configuration 3	3.5 Mbytes	1024 kByte	1.0 Mbytes	512 kByte

Table: configurable memory layout



### WARNING

When changing the preset memory layout.

- > Risk of personal injuries and/or damage to property.
- > Loss of the conditions for safety operation.
- ▶ For safety operation of the unit, use the preset configuration 2 (= MemoryLayout\_2\_25s\_2\_25).
- ▶ When changing the memory layout, safety operation is not permissible.

Description of the allocation of the memory layout in CODESYS: → **Assign memory allocation -- safety / standard PLC** (→ p. [78](#))

## 4.4.2 Device supply (technology)

### Content

Voltage ranges of the on-board system .....	42
Start conditions .....	43
Behaviour in case of voltage dip .....	43
Switch on/off via main switch .....	43
Switch on/off via ignition lock (clamp 15) .....	43
Sleep mode .....	45
Block diagram of the supply and of the output deactivation .....	45

39497

## Voltage ranges of the on-board system

39616

The system monitors the voltage ranges of the on-board system.  
The voltages mentioned here apply to the specified range of  $\pm 1\%$  (at 36 V).

Voltage [V]		Description
of	to	
	< +5.5	<b>Undervoltage</b> VBB15, VBB30: If the controller was in the OPERATING mode, the controller will shut down.
+5.5	< 8.0	<b>Limited operating range</b> If the controller was in the OPERATING mode, then it will continue to operate in this range without any restrictions if there are voltage dips.
+8.0	+32.0	<b>Regular operating voltage</b> Nominal operating voltage all functions available VBB15 > 5 V UND VBB30 > 8 V: The controller is booting
> +32.0	+36.0	<b>Overvoltage (protected)</b> The device is not damaged by the voltage deviation. If this condition on VBB15 / VBB30 in the OPERATING mode lasts longer than 10 s, the controller will change to the FATAL ERROR state. If this condition on VBB0...n in the OPERATING mode lasts longer than 10 s, the corresponding output group will change to the COMPONENT ERROR state.
> +36.0		<b>Overvoltage (unprotected)</b> In this area, the device is no longer protected and the behaviour is not predictable. The device can be destroyed by this voltage. ► If such voltages are likely to occur in an application, provide external protection!



### WARNING

Overvoltage > 36 V (unprotected)

- > Risk of personal injuries and/or damage to property.
- > Uncontrollable and unpredictable dangerous behaviour of the controller, e.g. all outputs activated.
- Provide external protection against overvoltage.

## Start conditions

22925

The device does not start before sufficient voltage is applied to supply connections VBB30 and VBB15 (= terminal 15).

In vehicles clamp 15 is the plus cable switched by the ignition lock.

This voltage must be provided by the on-board system of the mobile machine.

## Behaviour in case of voltage dip

60499

In the event of a voltage dip (e.g. due to motor start)  $VBB30 < 5.5 \text{ V}$ , the controller will shut down. If the VBB30 does not drop to 0 V, the control unit remains switched off after a recurring voltage  $VBB30 \geq 8 \text{ V}$ .

The behaviour in case of voltage dip can be changed by the application:

- ▶ Call the function Reset with RESET\_TYPE = POWER\_RECOVER once after starting the application. → **FUN Reset** (→ p. [214](#))
- > The controller executes an automatic power-on reset if  $VBB30 \geq 8 \text{ V}$ .

## Switch on/off via main switch

39607

To do so: VBB15 is connected with VBB30

### Procedure when switching on the main switch

- > The system recognises the applied voltage ( $VBB15 > 5 \text{ V}$  AND  $VBB30 > 8 \text{ V}$ ) and activates the connection of the controller to the VBB30 potential via solid-state switch.
- > The controller boots and starts.

### Procedure when switching off the main switch:

- > Running tasks will continue till the end.

39607



Data loss is possible in case of non-volatile data (retain data).

The tasks may not be longer than 50 ms!

Tasks that run for longer will be aborted before the end of the task.

- > the system automatically stores the retain data  
the input signals are no longer read  
the outputs are switched off.
- > the system switches off completely

If this behaviour is not wanted, use the circuit via ignition lock:

## Switch on/off via ignition lock (clamp 15)

39589

To do so:

- ▶ Connect the VBB15 via the ignition lock (Vehicle Clamp 15 \*) with the vehicle plus pole.
- ▶ Connect VBB30 directly with the vehicle plus pole (Vehicle Clamp 30).

\*) In vehicles clamp 15 is the plus cable switched by the ignition lock.

Latching can be configured on switch-on:

- ▶ Either:
  - set the input xValue of the FB **SupplySwitch** (→ p. [278](#)) to TRUE (latching activated, factory setting) or FALSE (latching deactivated)
- ▶ or:

- set the parameter [SUPPLY\_SWITCH] to the value TRUE/FALSE in the I/O configuration under [StandardPLC] > [Local\_IO] > [System\_Outputs].

**Procedure with switch-on via ignition lock with FB **SupplySwitch** (→ p. 278), input xValue = TRUE (factory setting):**

- > The ignition lock applies voltage to VBB15 (Vehicle Clamp 15 \*).
- > The system recognises the applied voltages (VBB15 > 5 V AND VBB30 > 8 V) and activates the connection of the controller to the VBB30 potential via solid-state switch.
- > When the application is started: The ignition switch is bypassed, latching of the control voltage is **activated**.
- > The controller boots and starts.

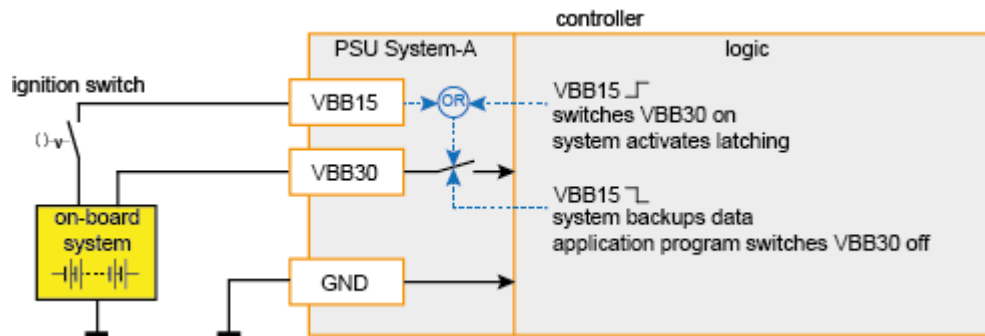


Figure: Delayed switch-off via ignition lock (clamp 15)

**Procedure with switch off via ignition lock with FB **SystemSupply** (→ p. 280), input xValue = TRUE (factory setting):**

- Evaluate VBB15 in the application via FB **SystemSupply** (→ p. 280)  
If VBB15 < 5 V:
  - execute necessary actions (e.g.):
    - stop machine gently
    - transmit required data
    - save required data and close
  - switch the input xValue of the FB **SystemSupply** (→ p. 280) from TRUE to FALSE
  - running tasks continue till the end.
  - stop the application
  - the system automatically saves the persistent data  
the input signals are no longer read  
the outputs are switched off.
  - the system deactivates the latching via VBB30:
  - the system switches off completely after the residual energy has been consumed

**Procedure with switch-on via ignition lock with FB **SystemSupply** (→ p. 280), input xValue = FALSE:**

- > The ignition lock applies voltage to VBB15 (Vehicle Clamp 15 \*).
- > The system recognises the applied voltages (VBB15 > 5 V AND VBB30 > 8 V) and activates the connection of the controller to the VBB30 potential via solid-state switch.
- > When the application is started: Latching of the control voltage is **deactivated**.
- > The controller boots and starts.

**Procedure with switch-off via ignition lock with FB **SystemSupply** (→ p. 280), input xValue = FALSE:**

- Evaluate VBB15 in the application via FB **SystemSupply** (→ p. 280)  
If VBB15 < 5 V:
  - running tasks continue till the end.



- stop the application
- the system automatically saves the persistent data  
the input signals are no longer read  
the outputs are switched off.
- the system switches off completely after the residual energy has been consumed

## Sleep mode

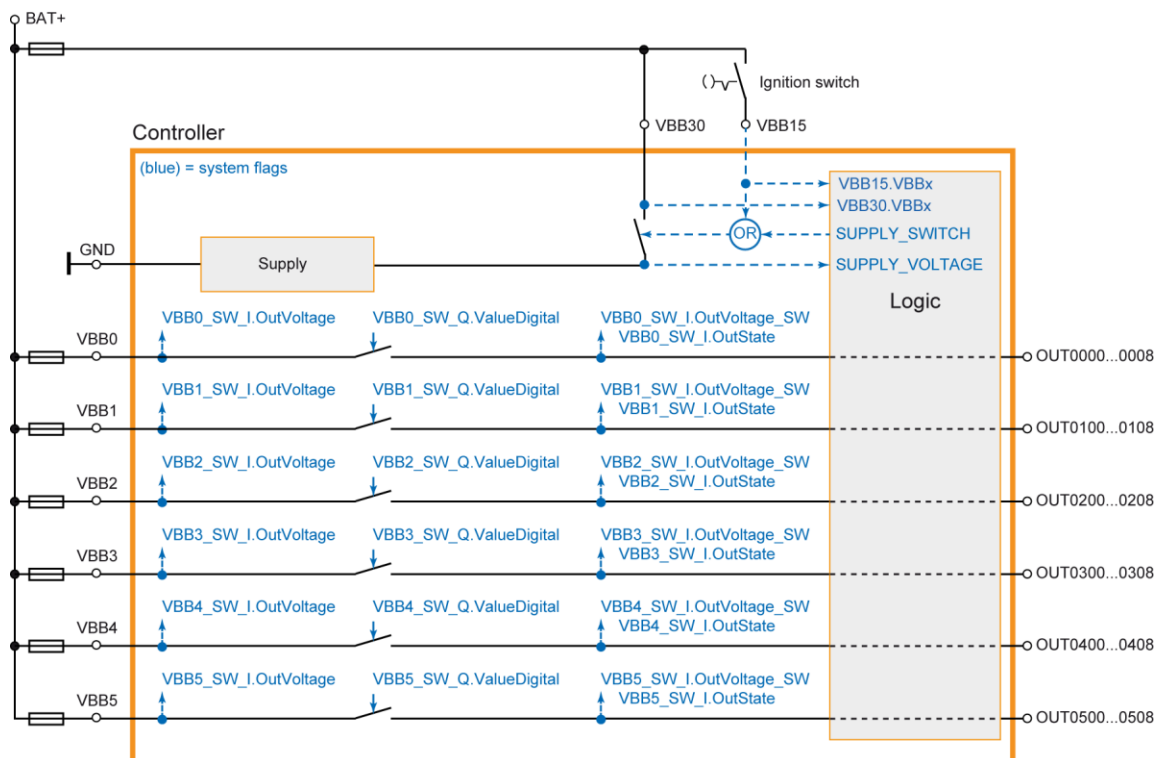
60500

The controller can be set to a power-saving mode by the IEC application if the VBB30 power supply is switched on and the VBB15 is switched on:

- The sleep mode is supported by devices with hardware version  $\geq$  V1.0.5.2.
- The hardware is almost switched off.
- Sleep mode is indicated by the blue SYS0 LED.
- After switching VBB15 off and on again, the control unit will reboot (PowerOn Reset).
- The sleep mode is activated with the **FUN Reset** (→ p. 214).

### Block diagram of the supply and of the output deactivation

58711



The image shows the block diagram of the supply and output deactivation for the device with maximum possible configuration: Some device versions have only some of the outputs shown.

## Switch off outputs via solid-state switch

58712

During the program process, the output switches are under the user's full software control.

If an error occurs during the program sequence, it is possible to disconnect the output switches from voltage via the FB **OutputGroup** (→ p. 299) in order to separate critical plant sections.



## **WARNING**

Unintentional and dangerous start of machine or plant sections.

- > Risk of personal injuries and/or damage to property.
- ▶ When creating the program, the programmer must ensure that the machine or plant sections cannot start without it being intended and constitute danger after an error occurs and is eliminated (e.g. E-stop).
  - Implement restart disable.
  - In case of an error, set the outputs concerned to FALSE in the program.

### 4.4.3 Inputs (technology)

#### Content

Cross references inputs.....	47
Types of inputs .....	48

28353

#### Cross references inputs

58713

Further information on the use of inputs:

For standard and safety PLC:

- **Configure inputs and outputs** (→ p. [105](#))
- **Options to access the input data and output data** (→ p. [112](#))
- **List of inputs** (→ p. [465](#))

For standard PLC:

- **Access inputs** (→ p. [118](#))

For safety PLC:

- **1-channel safety concept for inputs** (→ p. [140](#))
- **2-channel safety concept for inputs** (→ p. [150](#))

## Types of inputs

### Content

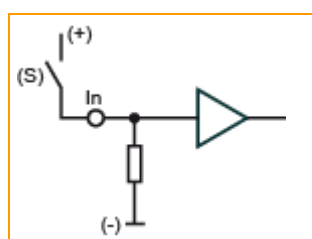
Digital input block diagram plus/minus-switching .....	48
Input type IN MULTIFUNCTION-A .....	48
Input type IN FREQUENCY-B .....	49
Input type IN RESISTOR-A .....	50
Input type IN DIGITAL-A/B .....	51

39633

We differentiate between the following input types:

### Digital input block diagram plus/minus-switching

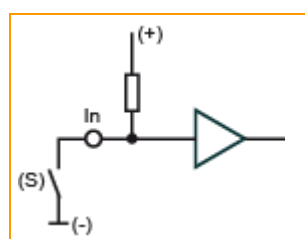
39506



**CSI = Current Sinking**

In = connection of digital input n  
(S) = sensor

Digital input block diagram, plus-switching ( $B_L$ )  
for positive sensor signal  
Input = open  $\Rightarrow$  Signal = Low (GND)



**CSO = current sourcing**

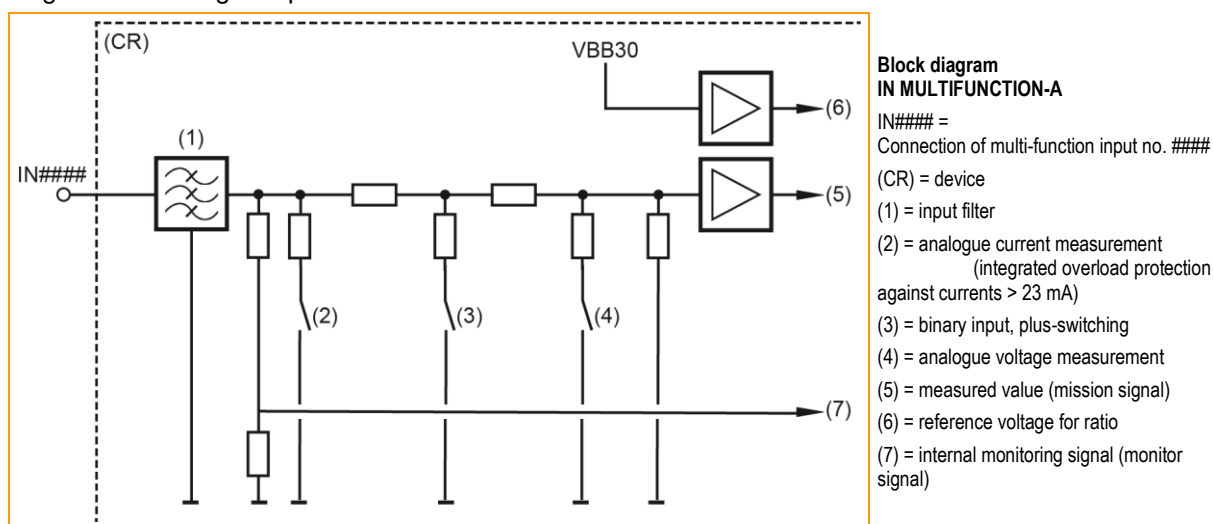
In = connection of digital input n  
(S) = sensor

Digital input block diagram, minus-switching ( $B_H$ )  
for negative sensor signal:  
Input = open  $\Rightarrow$  Signal = High (Supply)

### Input type IN MULTIFUNCTION-A

58714

Digital and analogue inputs



Input configuration  $\rightarrow$  Chapter **System configuration** ( $\rightarrow$  p. 75)

Possible operating modes		Technical data	Safety-compatible	
			1 channel	2 channels
IN_DIGITAL_CSI	Digital input	plus-switching ( $B_L$ )	X	X
IN_DIGITAL_CSI_NAMUR	Digital input	plus-switching ( $B_L$ ) for NAMUR sensors	X	X
IN_VOLTAGE_10	Analogue voltage measurement	0... 10 000 mV	X	X

Possible operating modes		Technical data	Safety-compatible	
			1 channel	2 channels
IN_VOLTAGE_32	Analogue voltage measurement	0...32 000 mV	X	X
IN_VOLTAGE_RATIO	Analogue voltage measurement, ratiometric to the reference voltage	0...1000 ‰	X	X
IN_CURRENT_CSI	Analogue current measurement	0...20 000 µA		X
Legend: <b>X = is</b> supported empty = is not supported				

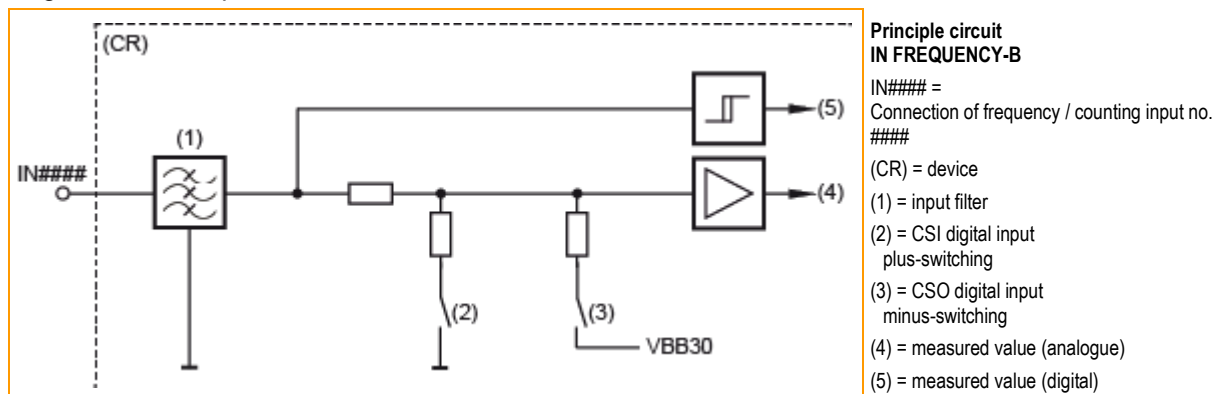


The signal of the measured value of an IN MULTIFUNCTION-A input (mission signal) is compared to the internal monitoring signal (monitor signal) and checked for plausibility. The plausibility check is **not** performed for IN MULTIFUNCTION-A inputs with operating mode IN\_CURRENT\_CSI.

## Input type IN FREQUENCY-B

60501

Digital and fast inputs



Input configuration → Chapter **System configuration** (→ p. 75)

Possible operating modes		Technical data	Safety-compatible	
			1 channel	2 channels
IN_DIGITAL_CSI	Digital input	plus-switching (B <sub>L</sub> )		X
IN_DIGITAL_CSO	Digital input	minus-switching (B <sub>H</sub> )		
IN_DIGITAL_CSI_NAMUR	Digital input	plus-switching (B <sub>L</sub> ) for NAMUR sensors		X
IN_DIGITAL_CSI_BLANKING	Digital input	plus-switching (B <sub>L</sub> ) Blanking signal for safe inputs	X	X
IN_VOLTAGE_10	Analogue voltage measurement	0...10 000 mV		X
IN_COUNT_CSI	Pulse counter	plus-switching (B <sub>L</sub> )		X
IN_COUNT_CSO	Pulse counter	minus-switching (B <sub>H</sub> )		
IN_FREQUENCY_CSI	Frequency measurement	plus-switching (B <sub>L</sub> )		X
IN_FREQUENCY_CSO	Frequency measurement	minus-switching (B <sub>H</sub> )		

Possible operating modes		Technical data	Safety-compatible	
			1 channel	2 channels
IN_INC_ENCODER_CSI	Evaluate the input pair for position encoder	plus-switching ( $B_L$ )		X
IN_INC_ENCODER_CSO	Evaluate the input pair for position encoder	minus-switching ( $B_H$ )		
IN_PERIOD_RATIO_CSI	Measuring of frequency, cycle time and pulse/pause ratio	plus-switching ( $B_L$ )		X
IN_PERIOD_RATIO_CSO	Measuring of frequency, cycle time and pulse/pause ratio	minus-switching ( $B_H$ )		
IN_PERIOD_RATIO_US_CSI	Measuring of frequency, cycle time and pulse/pause ratio	plus-switching ( $B_L$ )		X
IN_PERIOD_RATIO_US_CSO	Measuring of frequency, cycle time and pulse/pause ratio	minus-switching ( $B_H$ )		
IN_PHASE_CSI	Evaluate the input pair for the phase position	plus-switching ( $B_L$ )		X
IN_PHASE_CSO	Evaluate the input pair for the phase position	minus-switching ( $B_H$ )		
Legend: <b>X = is</b> supported empty = is not supported				

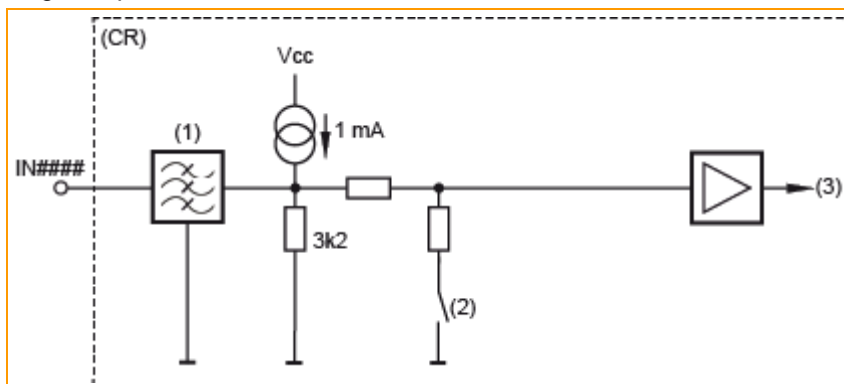


- The internal resistance  $R_i$  of the signal source must be substantially lower than the input resistance  $R_{input}$  of the input used (principle of voltage adaptation). Otherwise the input signal of the fast input can be distorted (lowpass characteristic).
- If operating mode IN\_DIGITAL\_CSO and input signal voltage > VBB30: The voltage measurement at the input becomes inaccurate and the overvoltage detection triggers early.

## Input type IN RESISTOR-A

39655

### Digital inputs and resistance measurement



**Block diagram  
IN RESISTOR-A**

IN#### =  
Connection of frequency / counting input no.  
####  
(CR) = device  
(1) = input filter  
(2) = CSI digital input  
plus-switching  
(3) = measured value (analogue)

Input configuration → Chapter **System configuration** (→ p. 75)

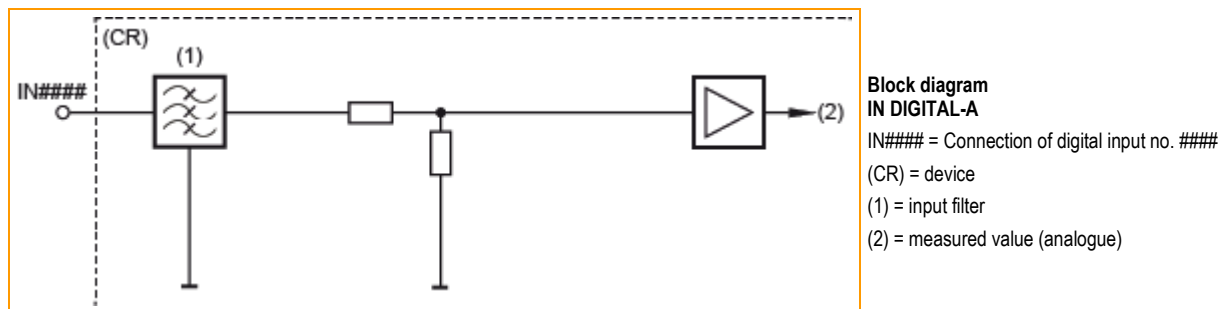
Possible operating modes		Technical data	Safety-compatible	
			1 channel	2 channels
IN_DIGITAL_CSI	Digital input	plus-switching ( $B_L$ )		X
IN_DIGITAL_CSI_NAMUR	Digital input	plus-switching ( $B_L$ ) for NAMUR sensors		X

Possible operating modes		Technical data	Safety-compatible	
			1 channel	2 channels
IN_RESISTOR	Resistance measurement (to VBB)	minus-switching ( $B_H$ )		
Legend: <b>X = is</b> supported empty = is not supported				

## Input type IN DIGITAL-A/B

39649

### Digital inputs



Input configuration → Chapter **System configuration** (→ p. [75](#))

Possible operating modes		Technical data	Safety-compatible	
			1 channel	2 channels
IN_DIGITAL_CSI	Digital input	plus-switching ( $B_L$ )		<b>X</b>
IN_DIGITAL_CSI_NAMUR	Digital input	plus-switching ( $B_L$ ) for NAMUR sensors		<b>X</b>
Legend: <b>X = is</b> supported empty = is not supported				

## 4.4.4 Outputs (technology)

### Content

Cross references outputs .....	52
Output types .....	53

28572

### Cross references outputs

58716

Further information on the use of outputs:

For standard and safety PLC:

- **Configure inputs and outputs** (→ p. [105](#))
- **Options to access the input data and output data** (→ p. [112](#))
- **List of outputs** (→ p. [471](#))

For standard PLC:

- **Access outputs** (→ p. [119](#))

For safety PLC:

- **Safety concept of the outputs** (→ p. [163](#))



## Output types

### Content

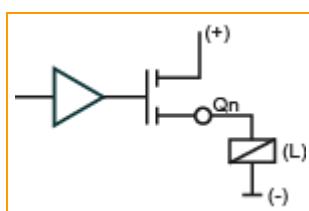
Digital output block diagram plus/minus-switching.....	53
Output type OUT PWM-n-A.....	53
Output type OUT PWM-n-B.....	54
Output type OUT PWM-n-BRIDGE-A.....	54
Output type OUT Supply-A.....	55
Output type OUT Voltage-A.....	56

39689

We differentiate between the following output types:

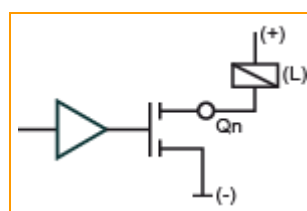
### Digital output block diagram plus/minus-switching

39516



**CSO = current sourcing**  
 $Q_n$  = connection of output n  
 (L) = load

Output block diagram plus-switching ( $B_H$ )  
for positive output signal



**CSI = current sinking**  
 $Q_n$  = connection of output n  
 (L) = load

Output block diagram, minus-switching ( $B_L$ )  
for negative output signal

### Output type OUT PWM-n-A

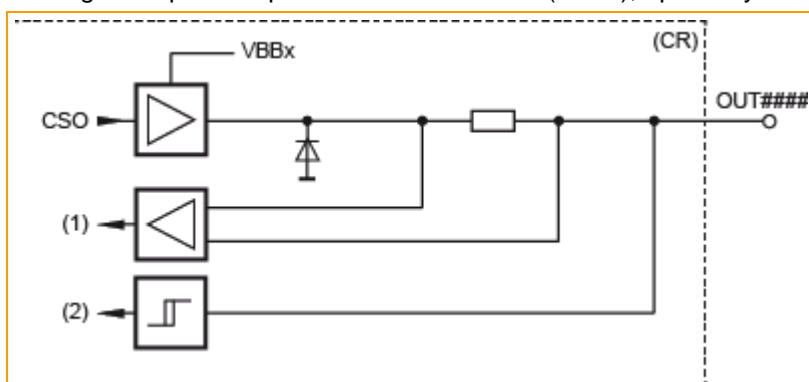
39678

n = current rating

Example:  $n = 25 \Rightarrow I_{max} = 2.5 \text{ A}$

Digital output or

analogue output with pulse width modulation (PWM), optionally current-controlled ( $PWM_I$ )



**Block diagram  
OUT PWM-n-A**  
 OUT#### = connection of PWM output no. ####  
 (CR) = device  
 (1) = measured value (analogue)  
 (2) = measured value (digital)

Output configuration → Chapter **System configuration** (→ p. 75)

Possible operating modes		Technical data
OUT_DIGITAL_CSO	Digital output	plus-switching ( $B_H$ ) fail-safe deactivation
OUT_PWM_CSO	analogue output CSO with pulse width modulation, without current measurement	$PWM_H$ plus-switching ( $B_H$ ) fail-safe deactivation
OUT_CURRENT_CSO	analogue output CSO with pulse width modulation, current-controlled	$PWM_I$ plus-switching ( $B_H$ ) fail-safe deactivation



The output diagnostics is internally limited to wire break, short circuit and overload.  
The project engineer has to carry out external measures to avoid cross fault in cabling.  
Reaction of the controller in case of an error: error class D, periphery error → **Error classes**  
(→ p. 462)

#### Setting and measurement via:

- FB **Output** (→ p. 273) for digital output
- FB **PWM1000** (→ p. 312) for PWM
- FB **CurrentControl** (→ p. 309) for current control (PWM<sub>I</sub>)

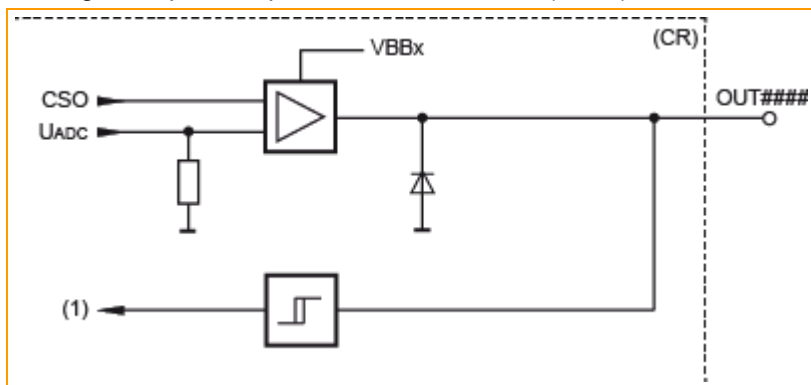
#### Output type OUT PWM-n-B

23093

n = current rating

Example: n = 25 ⇒  $I_{max} = 2.5 \text{ A}$

Digital output or  
analogue output with pulse width modulation (PWM)



Block diagram  
OUT PWM-n-B

OUT#### = connection of PWM output no. ####

(CR) = device

(1) = measured value (digital)

$U_{ADC}$  = current mirror

Output configuration → Chapter **System configuration** (→ p. 75)

Possible operating modes		Technical data
OUT_DIGITAL_CSO	Digital output	plus-switching ( $B_H$ ) fail-safe deactivation
OUT_PWM_CSO	analogue output CSO with pulse width modulation, without current measurement	PWM <sub>H</sub> plus-switching ( $B_H$ ) fail-safe deactivation



OUT PWM-n-B supports no dither functionality.  
The current measurement does not detect any free-wheeling current.

#### Setting and measurement via:

- FB **Output** (→ p. 273) for digital output
- FB **PWM1000** (→ p. 312) for PWM

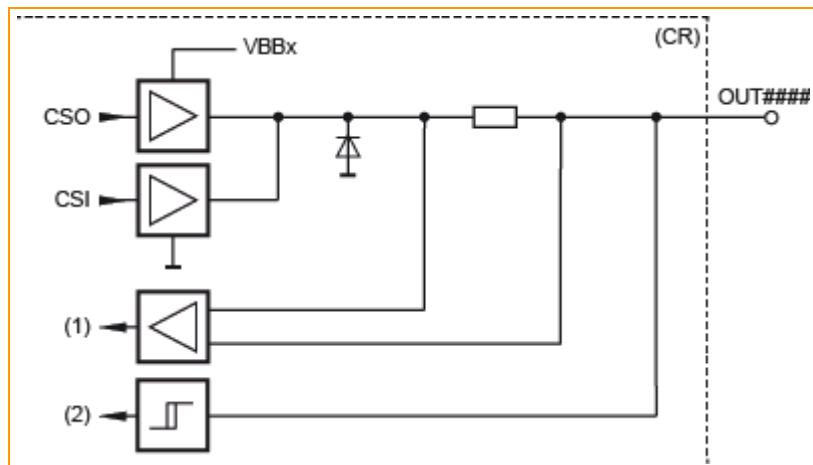
#### Output type OUT PWM-n-BRIDGE-A

39683

n = current rating

Example: n = 40 ⇒  $I_{max} = 4.0 \text{ A}$

Digital output or  
analogue output with pulse width modulation (PWM), optionally current-controlled (PWM<sub>I</sub>)  
or bridge output (via PWM)



**Block diagram**  
**OUT PWM-n-A**

OUT#### = connection of PWM output no. ####

(CR) = device

(1) = measured value (analogue)

(2) = measured value (digital)

Output configuration → Chapter **System configuration** (→ p. 75)

Possible operating modes		Technical data
OUT_DIGITAL_CSO	Digital output	plus-switching ( $B_H$ ) fail-safe deactivation
OUT_DIGITAL_CSI	Digital output	minus-switching ( $B_L$ )
OUT_PWM_CSO	analogue output CSO with pulse width modulation, without current measurement	$PWM_H$ plus-switching ( $B_H$ ) fail-safe deactivation
OUT_PWM_CSI	analogue output CSI with pulse width modulation, without current measurement	$PWM_L$ minus-switching ( $B_L$ )
OUT_CURRENT_CSO	analogue output CSO with pulse width modulation, current-controlled	$PWM_I$ plus-switching ( $B_H$ ) fail-safe deactivation

#### Setting and measurement via:

- FB **Output** (→ p. 273) for digital output
- FB **PWM1000** (→ p. 312) for PWM
- FB **CurrentControl** (→ p. 309) for current control ( $PWM_I$ )
- FB **HBridge** (→ p. 304) for bridge output

#### Output type OUT Supply-A

23125

The output OUT3000 is used to supply sensors with a stable voltage (5 V or 10 V) that is not affected by fluctuations of the supply voltage.

25101

### NOTICE!

External supply of the output.

> The output may be damaged.

► Do NOT apply any external voltage.

Possible operating modes		Technical data
OFF	Reference voltage output	0 V
OUT_SENSOR_05	Reference voltage output	5 V (to GND)
OUT_SENSOR_10	Reference voltage output	10 V (to GND)

Setting and measurement via FB **Output** (→ p. 273) or via system configuration:

## Setting /measurement via FB Output

25098

### Setting the sensor supply:

- Use the inputs in the FB output as follows:  
     [uiChannel] = 3000  
     [eMode] = [OUT\_SENSOR\_05] (5 V) or  
     [eMode] = [OUT\_SENSOR\_10] (10 V)

### Monitoring the values at the sensor supply output:

- Read the outputs in the FB output as follows:
- > [uiOutVoltage] indicates the measured voltage in [mV]  
     [uiOutCurrent] indicates the measured current in [mA]



Do not use the value [uiOutCurrent]. (great inaccuracy).

Details → FB **Output** (→ p. [273](#)).

## Output type OUT Voltage-A

25028

The output provides 0...10 V e.g. for further controllers or actuators.

CR71nS: only OUT3001

CR72nS: OUT3001 and OUT3002

The output is protected against overload and will switch off automatically if overloaded.

25101

### NOTICE!

External supply of the output.

- > The output may be damaged.
- Do NOT apply any external voltage.

Possible operating modes		Technical data
OUT_ANALOG_10	analogue output	0...10 000 mV (to GND)

Setting and measurement via FB Output or via system configuration:

## Setting /measurement via FB Output

25103

### Setting the analogue output:

- Use the inputs in the FB output as follows:  
     [uiChannel] = 3001 / 3002  
     [uiValue] = required voltage in [mV]  
     permissible = 0...10000

### Monitoring the values at the analogue output:

- Read the outputs in the FB output as follows:
- > [uiOutVoltage] indicates the measured voltage in [mV]  
     [uiOutCurrent] indicates the measured current in [mA]

Details → FB **Output** (→ p. [273](#)).

#### 4.4.5 Feedback in case of externally supplied outputs

39509



**Do not apply any external voltage to the outputs!**

- > As soon as output group switch VBBn\_SW\_Q = FALSE:  
The internal device monitoring checks the voltage on the contact bar after the output group switch.  
If then a voltage of > 0.4 VBBn is measured:
- the controller reports error class C,
  - the controller switches the group to the safe state.



 Safe state of the group = all outputs are switched off

- all outputs will be switched off
- the controller reports the error to the IEC application

To reboot the device:

- ▶ remove the error cause
- ▶ do a power-on reset.

OR error handling in the IEC application:

- ▶ remove the error cause
- ▶ Remove error of the group via xResetError

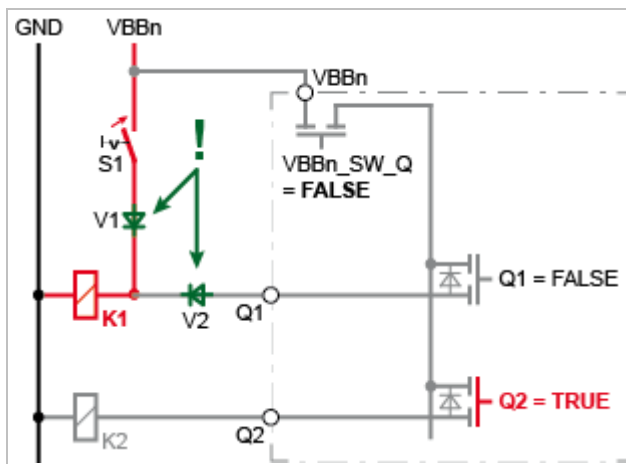


Figure: Example of wiring with blocking diodes due to the risk of feedback

### Remedy:

Insert the blocking diodes V1 and V2 (→ green arrows)!

**Successful:**

If VBBn\_SW\_Q = FALSE, the controller does not go to error class C when the S1 contact is closed.

**NOTE**

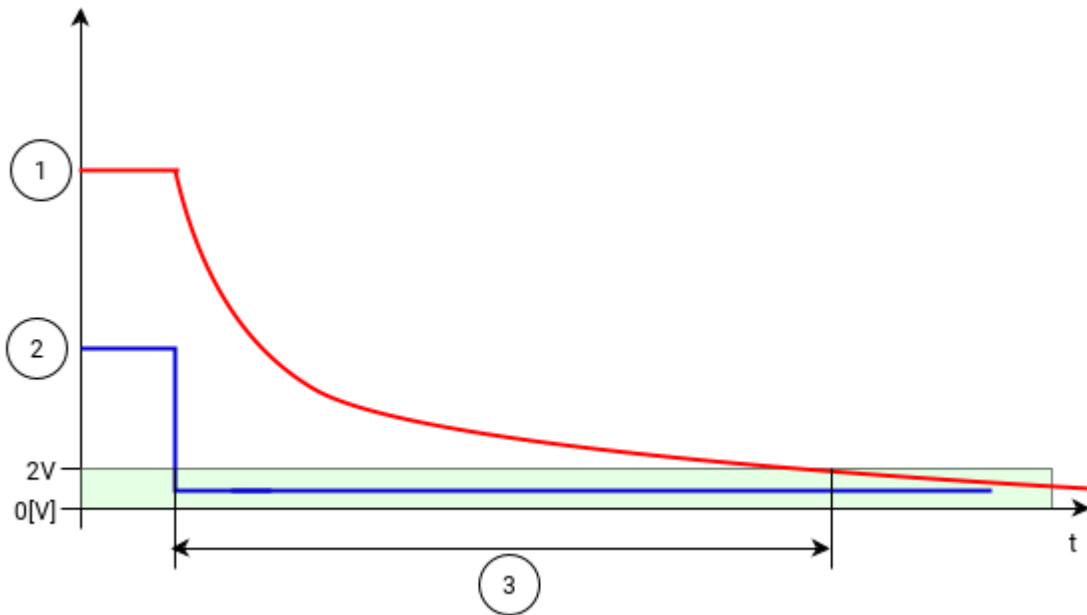
## Help for externally connected outputs

- ▶ Decouple the externally connected outputs by means of diodes so that no external voltage can be connected to the output terminal of the controller!

#### 4.4.6 Influence of actuators on the output diagnostics

58717

The voltage at the output must have fallen below the threshold value of the diagnosis status feedback 10 ms (standard value of the DetectionTime) after switch-off of the output (→ data sheet). Otherwise a STUCK\_AT\_HIGH error is triggered.



Capacitive or inductive loads can trigger a STUCK\_AT\_HIGH error at switch-off of an output (2) if the discharge (1) lasts longer than the set DetectionTime (3). As a result, the entire output group switches off.

- Use the function block **ConfigDiagLevel** (→ p. [290](#)) to set the DetectionTime (3) of the output to a value suitable for the load.

## 4.5 Interfaces

### Content

Serial interface.....	59
Ethernet interface .....	59
CAN: Interfaces and protocols .....	60

23132

The device includes the interfaces described in the following.



Position of the connections on the device and technical data: → Operating instructions / data sheet

### 4.5.1 Serial interface

39599

This device features a serial interface.

Connections and data → data sheet

### 4.5.2 Ethernet interface

60504

#### NOTICE!

If the device is operated in an unprotected network environment.

- > Unauthorised data access (read or write) is possible.
- > Unauthorised manipulation of the device functions is possible.
- Check and restrict access options to the device:
  - Restrict access to authorised persons.
  - Do not connect the device to open networks or the internet.
  - If access from the internet should be necessary, it is mandatory to choose a secure method to connect the device (e.g. VPN).

This unit provides a 2-port Ethernet interface via an internal switch. This enables line cabling between several devices.



The Ethernet interface supports the following standards:

- Data rate 10/100 MBit/s

The Ethernet interface supports the following protocols:

- TCP/IP
- UDP/IP
- Modbus/TCP Slave (in preparation)
- Modbus/TCP Master (in preparation)
- Network variables UDP

Connections and data → data sheet

### 4.5.3 CAN: Interfaces and protocols

24512



► Familiarise yourself with the following CODESYS functions!

- CAN-based fieldbuses  
→ Online help > Fieldbus support > CAN-based fieldbuses

The device has 4 independent CAN interfaces. Each CAN interface supports the following protocols:

- RawCAN (CAN Layer 2)
- CANopen Manager
- CANopen Device
- CANopen Safety Manager
- CANopen Safety Device
- J1939 Manager



4.6 Software description

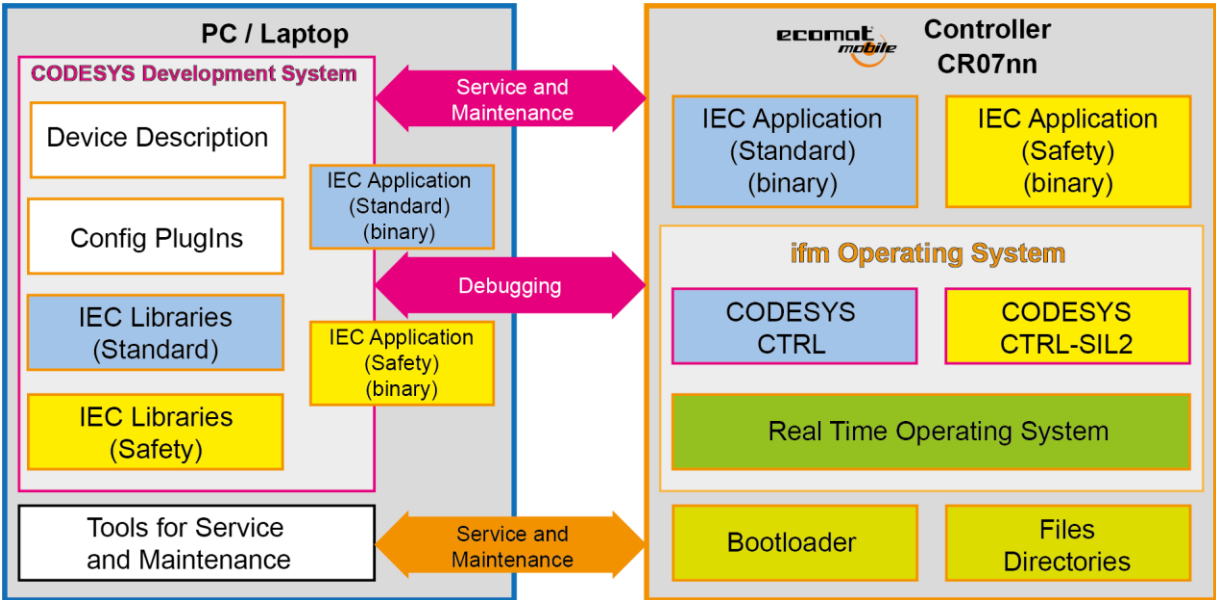
Content	
Overview: Software .....	61
Software module for the device .....	62

39606

4.6.1 Overview: Software

39676

We differentiate between the following software components:



Software on the PC/notebook.

39603

The programming environment CODESYS Development System is installed in the PC/notebook to create and debug both applications. The controller supports service and maintenance via CODESYS or via other tools.

The CODESYS functions are extended with Config plugins. Thereby, additional setting options for memory and inputs/outputs become available. ifm electronic provides adequate device descriptions for the CODESYS Development System for each derivative. IEC libraries for the safe and non-safe applications provide CODESYS and the programmer with access options to the functions of the controller.

Licensing

39639

By buying a controller CR7xxS, the buyer also purchases a licence that is valid for the use of the CODESYS 3.5 programming system.

Software in the controller

24514

The controller processes the applications by means of several software components. The ifm operating system with the CODESYS CTRL-SIL2 and the CODESYS CTRL constitutes, among other things, the processing time environment that executes both applications. The real-time

operating system enables separate execution of the safe and the non-safe software components in the controller.

Using the configuration file `comconf.cfg`, the programmer can control the interface.

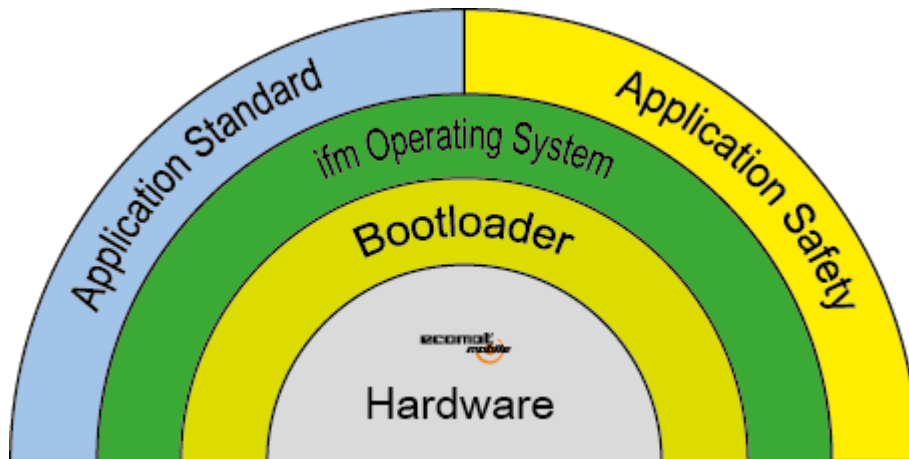
The programmer can store files and directories on the controller and use them in the application. Or the applications themselves create files and store them in the controller.

The bootloader is a fallback level for cases in which the ifm operating system is no (longer) available or it is corrupt.

## 4.6.2 Software module for the device

39602

The software in this device communicates with the hardware as follows:



Software module	Can the user change the module?	Using what?
Application with libraries a) for standard PLC b) for safety PLC	yes	CODESYS, Tools for service and maintenance
ifm operating system	Upgrade yes Downgrade yes	CODESYS, Tools for service and maintenance
Bootloader	no	---
(Hardware)	no	---

We describe this software module in the following:

### Bootloader

25105

The bootloader is a start program with which the operating system can be reloaded on the device.

54276



Only execute the bootloader update when explicitly requested by ifm!

## Operating system

39680

Basic program in the device, establishes the connection between the hardware of the device and the application.

→ Chapter **Software module for the device** (→ p. [62](#))

The device is supplied with the installed operating system.

Verifying and changing the operating system version → Chapter **Check the operating system version of the device** (→ p. [181](#))

The operating system only needs to be downloaded once - if at all. The application can then be loaded (also several times) in a PLC without affecting the operating system.

The operating system can be downloaded from **ifm electronic gmbh**'s website:

→ [www.ifm.com](http://www.ifm.com)

## Application

39572

Software specific to the application, implemented by the machine manufacturer, generally containing logic sequences, limits and expressions that control the appropriate inputs, outputs, calculations and decisions.



### WARNING

The application programs do not function properly or the configuration is incorrect.

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function.
- ▶ The system manufacturer is responsible for the functioning of the application programs.
- ▶ For applications of functional safety, the system manufacturer must assure the conformity of the system and the corresponding application programs in accordance with the applicable regulations.
- ▶ Certification by competent organisations may be required.

## Libraries

54278

ifm electronic provides the following function libraries for the programming of the device under CODESYS 3.5:

Name	Description
ifmCANOpenManager	Functions for the use of the CAN interfaces as CANopen Manager
ifmDeviceCR7xxS	Data structures, enumeration types and global variables
ifmFastInput	Functions to access the fast inputs of the device
ifmIOcommon	Functions for access to the inputs and outputs of the device
ifmIOconfigDiagProt	Functions to configure the I/O-related diagnostic and protective functions
ifmOutGroup	Functions to control output group switches
ifmOutHBridge	Functions to access H-bridge outputs
ifmOutPWM	Functions to access PWM outputs
ifmRawCAN	Functions for use of the CAN interfaces as CAN Layer 2
ifmSysInfo	Functions to set / read system information
ifmTypes	Global types and interfaces for other ifm libraries



Detailed information about the ifm function libraries: → **ifm function libraries** (→ p. [207](#))

## Libraries for safety programming

24516  
24548

For safety programming of the device, **ifm electronic** provides the following certified function libraries under CODESYS 3.5. The function blocks are specially designed for programming at basic and extended level.



Detailed information about the levels of complexity: → **Complexity level / user level** (→ p. [131](#))

Name	Description
ifmPLCopenSafe	Safety standard functions according to PLCopen specification
ifmIOSafety	Additional safety IO functions
ifmPLCopenAddOnSafe	Additional safety functions to use input signals with standard data types in safety applications



Detailed information about the **ifm** function libraries: → **ifm function libraries** (→ p. [207](#))

## 5 Installation

### Content

System requirements.....	65
Carry out installation.....	66

39661

This chapter describes the installation of the software components that are necessary to program the CR7xxS.

### 5.1 System requirements

#### Content

Hardware .....	65
Software.....	65

39622

Under which conditions can and may this device be programmed and operated?

#### 5.1.1 Hardware


22912

- Device from the **ifm** product family ecomatController
- PC/laptop for CODESYS programming system  
(→ Chapter Software > System requirements CODESYS Development System V3.5)
- Ethernet connection between CODESYS PC/laptop and Ethernet interface of the device  
(→ Operating instructions)

#### 5.1.2 Software

24843

To configure and program the device-internal standard PLC and the safety PLC of the CR7xxS, the following software components are required:

Component	of	Description	Version
CODESYS Development System	CODESYS	Programming software CODESYS for PLC programming according to standard IEC 61131-3	V3.5 SP11  The following applies to the safety PLC: ► Use the indicated CODESYS version without additional patches!
[CR7xxS]_Safety_V3.1.x.y.zip	ifm	Software package for ecomatController CR7xxS, consisting of: → <b>Components of the software package</b> (→ p. 66)	V3.1.x.y



The features and functions described in this manual can only be obtained by using the software components in the versions stated here.

The software components can be downloaded on the ifm electronic website:

→ [www.ifm.com](http://www.ifm.com)

The software package can only be downloaded after registration.

## 5.2 Carry out installation

### Content

CODESYS Development System .....	66
Complete package for ecomatController CR7xxS .....	66

39476

### 5.2.1 CODESYS Development System

39481

The CODESYS Development System (short: CODESYS) is a platform for the creation of PLC applications according to the standard IEC 61131-3.

#### Install CODESYS Development System

39659

To install the software "CODESYS Development System":



For installation on the PC/laptop, administrator rights are required.

- Install the programming system CODESYS V3.5 SP11.  
→ CODESYS installation and first steps
- > CODESYS V3.5 SP11 is installed on the PC/laptop.

### 5.2.2 Complete package for ecomatController CR7xxS

39482

#### Components of the software package

24846

ifm offers a software package to program the device-internal standard PLC and the device-internal safety PLC. The package contains the following components:

Data name / path	Description
<b>CR7xxS_[Version].zip</b>	Software package (ZIP file). The version follows the version of the operating system.
- ReleaseNotes.pdf	Release notes as PDF
+ <b>CODESYS_Package</b>	Folder for corresponding CODESYS device packages and plugins
- ifm_ecomatOS_[Version].package	CODESYS device package CR7xxS (device description, libraries, etc.)
+ <b>Plugin</b>	Folder with the CODESYS plugins for SIL2 and ifm SIL2 extensions.
- CODESYS Safety SIL2 xyz.package	Package "CODESYS Safety SIL2 Plugin"
- ifm Safety SIL2 Extensions Vn.n.n.n.package	Package "ifm Safety SIL2 Plugin"
+ <b>Device</b>	Folder for corresponding files that can be loaded to the device.
+ <b>boot</b>	Folder for the bootloader
- boot.ifm	Bootloader binary file
+ <b>compat</b>	Folder for the compatibility file

- CR7xxS-compat.ifm	Compatibility file
+ <b>os</b>	Folder for the operating system
- ifmOS.ifm	Operating system binary file
+ <b>cfg</b>	Folder with default configuration files
- comconf.cfg	Communication configuration (default)
- CR7xxS-iomapping.cfg	Input / output resources (default)
- CR7xxS-memconf-1024S-3584.ifm	Memory configuration 1
- CR7xxS-memconf-2304S-2304ifm	Memory configuration 2 (default)
- CR7xxS-memconf-3584S-1024.ifm	Memory configuration 3
+ <b>sis</b>	Folder for hardware description
- CR7xxS-[HW-VERSION]-sisys.ifm	One file per supported hardware version

## Install package (PC/laptop)

39660

To install a package

### Requirements

- > CODESYS V3.5 SP11 is installed on the PC/laptop.
- > ifm package "CODESYS for ifm R360III products" is stored on the PC/laptop.

#### 1 Start CODESYS

- ▶ Start CODESYS as administrator.
- > CODESYS user interface appears.

#### 2 Start Package Manager

- ▶ Select [Tools] > [Package Manager] to start the Package Manager.
- > Package manager appears.
- > Window shows installed packages.

#### 3 Install package

- ▶ Click on [Install...].
- > The file explorer appears.
- ▶ select the required file\*.package and carry out a full installation.
- > The [Package Manager] window shows the installed package.
- ▶ Click on [Close] to quit the Package Manager.
- ▶ Close CoDeSys
- ▶ Start CODESYS
- > The installed package is now available.

To install another package, proceed again as described.

## Update package (PC/laptop)

24823

To update a package:

#### 1 Uninstall the old version of the package

- ▶ **Uninstall package (PC/laptop)** (→ p. [68](#))

#### 2 Install the new version of the package

- ▶ **Install package (PC/laptop)** (→ p. [67](#))

#### 3 Update device

- ▶ In the device tree: Highlight [CR7xxS (CR7xxS)] node.
- ▶ Select [Project] > [Update Device...]

- > A dialogue window appears.
- ▶ Click on [Update Device] to start the update process.
- > CODESYS loads new device libraries.
- > Device tree view is updated.
- ▶ Click on [Close] to quit the Package Manager.
- ▶ Save the project.

## **Uninstall package (PC/laptop)**

39634

To uninstall a package:

### **1 Start package manager**

- ▶ Select [Tools] > [Package Manager] to start the Package Manager.
- > Window [Package Manager] shows installed packages.

### **2 Uninstall package**

- ▶ Activate checkbox [Display versions].
- > The window shows the version numbers of the installed packages.
- ▶ Select the package version to be uninstalled and uninstall it with [Uninstall...].
- > Selected package version is uninstalled.
- ▶ Click on [Close] to quit the Package Manager.



## 6 Getting started

### Content

Use the CODESYS operating instructions .....	69
Start CODESYS .....	69
Create CODESYS project .....	70
Configure the programming interface .....	73
Activate the access protection for a project.....	74

39512

### 6.1 Use the CODESYS operating instructions

24702  
54298

This manual only describes the integration, configuration and the programming of the CR7xxS using the CODESYS development system.

For the description of user actions and user interface elements the CODESYS terminology will be used.

Standard functions and methods of CODESYS will not be described. At the beginning of each section there will be a reference to the corresponding chapters of the CODESYS online help.

To access the online help of the CODESYS development system:

- ▶ Start CODESYS.
- > The CODESYS user interface appears.
- ▶ Press [F1].
- > Online help of the CODESYS development system appears.



- ▶ Familiarise yourself with the CODESYS development system! In particular with the following topics:

- Names and functions of the user interface elements
- Basic menu functions
- Programming techniques and methods for data retention



- ▶ Additionally familiarise yourself with the following topic to program safety applications:
  - Compound Safety PLC and CODESYS Safety SIL2

### 6.2 Start CODESYS

39583

#### Requirements

- > Software components are correctly installed (→ **Carry out installation** (→ p. [66](#))).

#### Start CODESYS

- ▶ Double-click on [CODESYS V3.5 SP11] symbol
- > CODESYS starts.
- > CODESYS user interface appears.

## 6.3 Create CODESYS project

### Content

Create new project with CR7xxS.....	71
Overview: Project structure with CR7xxS .....	72

39501



► Familiarise yourself with the following CODESYS functions!

- Create a project  
→ Online help > CODESYS Development System > Creating and Configuring a Project
- Manage a project  
→ Online help > CODESYS Development System > Protecting and Saving the Project

ifm electronic provides a special template for each model of the device family. The user can select the corresponding template when the project is created.

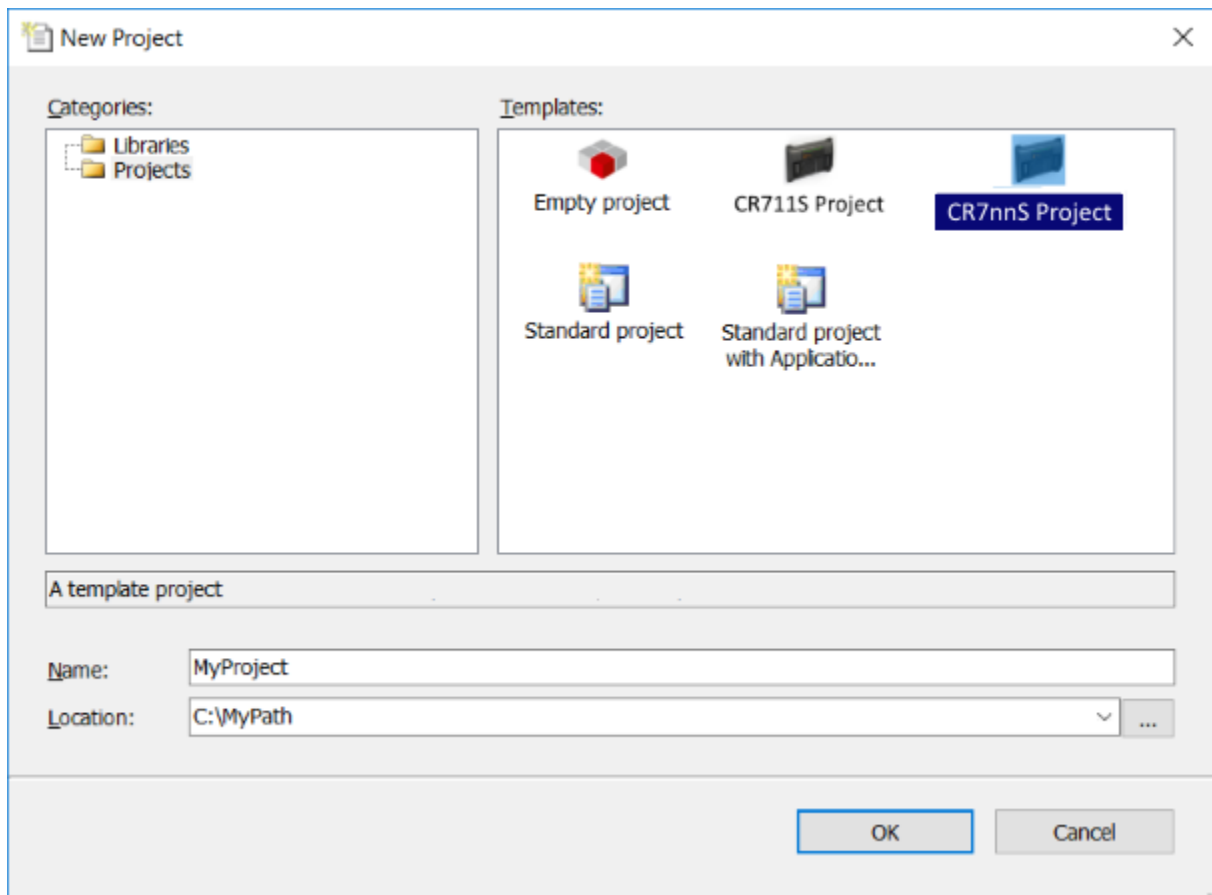
### 6.3.1 Create new project with CR7xxS

#### Requirements

- > ifm package "CODESYS for ifm R360III products" has been correctly installed (→ **Carry out installation** (→ p. 66)).

#### 1 Create new CR7xxS project

- ▶ Select [File] > [New Project...].
- > The window [New Project] appears.



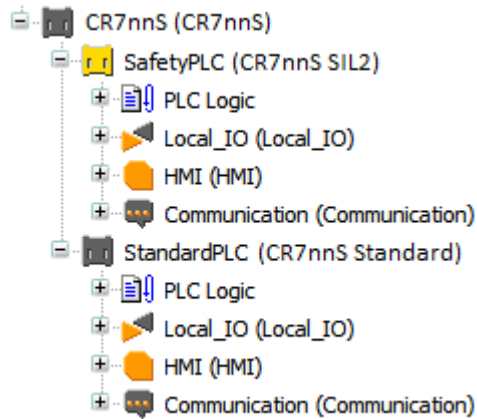
- ▶ Set the following values:
  1. [Templates]: Select the device project template, e.g. [CR0721 Project]
  2. [Name]: Enter the project name
  3. [Location]: Select storage location for the project file
- ▶ Click on [OK] to adopt the selected values.
- > CODESYS creates a new CR7xxS project.
- > The window [Devices] shows the device tree of the project (→ **Overview: Project structure with CR7xxS** (→ p. 72)).

#### 2 Save the project

- ▶ Select [File] > [Save Project].
- > CODESYS saves the project.

## 6.3.2 Overview: Project structure with CR7xxS

A CODESYS project contains all components to configure, manage and program the CR7xxS. All components of a project are shown in the window [Devices] in a hierarchic tree view. CODESYS projects with a CR7xxS have the following structure:



Legend:

CR7xxS (CR7xxS)	Logical father controller, offers access to the general settings of the CR7xxS → <b>Configuring the standard PLC</b> (→ p. <a href="#">85</a> )
SafetyPLC (CR7xxS SIL2)	Content of the SaFeTyPLC, offers access to the settings of the safety PLC
StandardPLC (CR7xxS Standard)	Content of the StandardPLC, offers access to the settings of the standard PLC
PLC logic	contains the applications of the CR7xxS → <b>Objects of a standard PLC application</b> (→ p. <a href="#">108</a> )
System_Info	provides access to the device information → <b>Display system information</b> (→ p. <a href="#">195</a> )
Local_IO	provides access to the configuration options of the inputs and outputs → <b>Configure inputs and outputs</b> (→ p. <a href="#">105</a> )
HMI	provides access to the configuration options of the operating and display elements
Communication	provides access to the configuration options of the communication interfaces → <b>Configuring CAN interfaces</b> (→ p. <a href="#">96</a> )

The programmer can adjust the terms in the structure before the expression in brackets:



- ▶ Right mouse click on the term > [Properties...]
- > The window [Properties] appears > tab [Common]
- ▶ enter a term
- ▶ confirm with [OK]

## 6.4 Configure the programming interface

### Content

Set communication path of PLC .....	73
Check the communication path (blinking test) .....	74

23495

The device-internal PLC can be programmed via the Ethernet interface of the device (position of the connections: → operating instructions).



Device and PC/laptop can be coupled directly or indirectly via an Ethernet network.

- ▶ Only use the recommended accessories for connection of the Ethernet interfaces! (→ operating instructions)
- ▶ For network connection, an experienced user or system administrator should set up the network addresses and the configuration.

### NOTICE!

If the device is operated in an unprotected network environment.

- > Unauthorised data access (read or write) is possible.
- > Unauthorised manipulation of the device functions is possible.
- ▶ Check and restrict access options to the device:
  - Restrict access to authorised persons.
  - Do not connect the device to open networks or the internet.
  - If access from the internet should be necessary, it is mandatory to choose a secure method to connect the device (e.g. VPN).

### 6.4.1 Set communication path of PLC

39594

To configure the communication path between the programming system CODESYS and the device-internal PLC:

#### Preparations

- > CODESYS PC/laptop and Ethernet interface of the device are connected.
- > Optional: Adjust IP settings of the Ethernet interface.

#### 1 Select communication settings

- ▶ In the device tree: Double-click on symbol [Device (CR7xxS)]
- > In the editor window: Select tab [Communication].
- > Editor window shows communication settings.

#### 2 Select gateway

- ▶ Select the requested gateway in the list [Gateway].
- > List shows selected gateway.

#### 3 Set communication path

- ▶ Activate [Scan Network ...].
- > Window [Select Device] appears.
- ▶ Select gateway node and start scan process with [Scan network].
- > CODESYS scans network for devices.
- > Window shows network path and detected devices.

- ▶ Select node of the device and activate [OK] to set the communication path to the device-internal PLC.
- > CODESYS can transfer data to the device-internal PLC.

### 6.4.2 Check the communication path (blinking test)

25068

To check the set communication path, execute a blinking test of the device:

- ▶ In the device tree: Double-click on the symbol [Device (CR7xxS)]
- ▶ In the editor window: Select tab [Communication Settings].
- ▶ Enable [Scan Network...].
- > Window [Select Device] appears.
- ▶ Select the gateway node and start the scan process with [Scan network].
- > CODESYS scans the network for devices.
- > Window shows the network path and detected devices.
- ▶ Select the node of the device
- ▶ Click on the [Wink] button
- > The SYS0-/SYS1-LEDs of the connected device flash in a colour sequence when the connection is established.  
The colour sequence is: green-red-yellow, 2 seconds per colour and an overall duration of 20 seconds.

## 6.5 Activate the access protection for a project

39569



- ▶ Familiarise yourself with the following CODESYS functions!
  - Protect and save project
    - Online help > CODESYS Development System > Protect and save project

The user can use a password to protect the device from unauthorised access.

- ▶ Select [Project] > [Project Settings...].
- > Window [Project Settings] appears.
- ▶ Select [Security].
- ▶ Activate checkbox [Enable project file encryption].
- ▶ Enter the requested password in the field [New password].
- ▶ Enter the entered password again in the field [Confirm new password].
- ▶ Select [OK] to activate the access protection for the project.
- > Access protection is activated. Project is encrypted.

## 7 System configuration

### Content

Preparation of the PLC .....	75
Configuring the standard PLC .....	85
Configuring the safety PLC.....	88
Use memory .....	90
Configure IEC watchdog.....	93
Configure interfaces .....	94
Configure inputs and outputs .....	105

39625

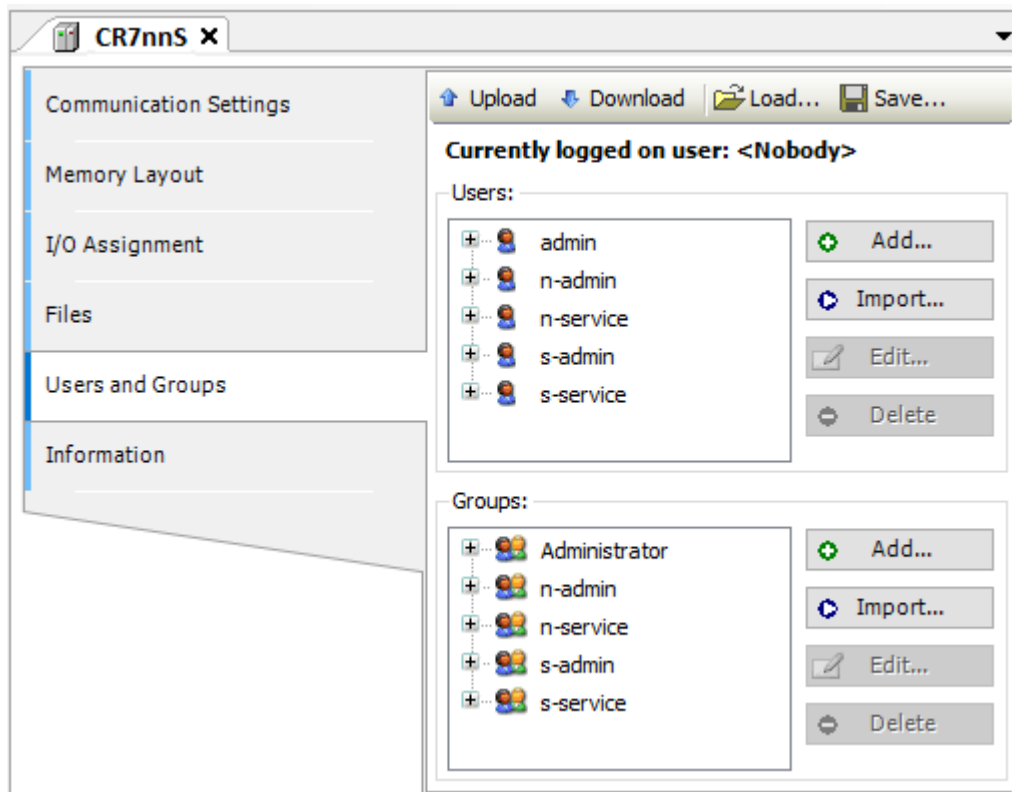
### 7.1 Preparation of the PLC

#### Content

Managing device users.....	76
Create a user in the CODESYS project .....	77
Sign the user in to the CODESYS project.....	78
Enter information about applications .....	78
Assign memory allocation -- safety / standard PLC .....	78
Assign inputs/outputs - safety / standard PLC .....	79
Manage files .....	81
User-defined data .....	83

24518

## 7.1.1 Managing device users



### 1 To open the user configuration:

- ▶ In the CODESYS device tree: Double-click on the symbol [CR7xxS (CR7xxS)]
- ▶ In the editor window: Select the tab [Users and Groups].
- ▶ Click on the [Upload] button.
- > The user configuration is loaded from the controller.
- > The editor window shows the user configuration.
- > The following users are available:
  - admin
  - n-admin
  - n-service
  - s-admin
  - s-service



Information about the user authorisations:  
→ **Overview of user rights** (→ p. [487](#))



By default, the users do not have a password.  
The project engineer must assign a password to each user to prevent unauthorised access to the project and the controller.

### 2 Assigning a password

- ▶ In the CODESYS device tree: Double-click on the symbol [CR7xxS (CR7xxS)]



- ▶ In the editor window: Select the tab [Users and Groups].
- ▶ Click on the [Load] button
- > The editor window shows the user configuration.
- ▶ Double-click on one of the shown users
- > The window [Edit] appears
- ▶ Enter the password twice (maximum 32 characters)
- ▶ Close the window by clicking [OK]
- ▶ After the change: Load the user data to the controller with [Download]
- ▶ Restart the controller
- ▶ Save changes with: Click [File] > [Save Project]



(User) groups are not supported.



For this action, you need to sign in as device user.

For more user convenience:

- ▶ Create a user in the project and sign the user in once
  - → **Create a user in the CODESYS project** (→ p. [77](#))
  - → **Sign the user in to the CODESYS project** (→ p. [78](#))

## 7.1.2 Create a user in the CODESYS project

25089

To ensure that the user data does not need to be entered again when working with the CODESYS project, store it in the project:

25131



- ▶ Make sure to use the correct lower-case and upper-case letters.

- ▶ Menu [Project] > [Project Settings] > [Users and Groups] > [Add...]
- ▶ Enter:
  - [Login name:] s-admin
  - [Full name:] s-admin
  - [Description:] Safety Administrator (e.g.)
  - [Password:] your password (must be identical with the password of the device user s-admin)
  - [Password:] repeat your password
  - [Activate:] enable
  - Click [OK]
  - If you have not yet done so, sign in as Owner to be able to create the new user (→ **Sign the user in to the CODESYS project** (→ p. [78](#))).
- > The user has been created in the CODESYS project.

### 7.1.3 Sign the user in to the CODESYS project

25115

Sign in to the project after saving the user details in the CODESYS project:

25131



- ▶ Make sure to use the correct lower-case and upper-case letters.

- ▶ Double-click on [Current User: ...] in the status bar
  - > The user change dialogue appears
- ▶ Click [Logon as another user...]
- > The login dialogue appears
- ▶ Sign in as Owner:
  - [User name:] Owner
  - [Password:] no password
  - Click [OK]
- ▶ Double-click again on [Current User: ...] in the status bar
- ▶ Sign in as s-admin:
  - [User name:] s-admin
  - [Password:] the corresponding password
  - Click [OK]
- > Login as s-admin is taking place.
- > It is necessary to sign in one more time after restarting the project.
- > The login details will be stored.

### 7.1.4 Enter information about applications

25093

- ▶ To identify and easily differentiate between the standard application and the safety application, enter useful information for both applications as follows:
- ▶ Right mouse click on [Application] > [Properties] > [Information] tab
- ▶ Enter useful texts for:
  - [Author]
  - [Version]
  - [Description], e.g. safety PLC control cabinet front or standard PLC control cabinet rear
- ▶ Close the window by clicking [OK]
- ▶ Save the project with [File] > [Save Project]
- > After translating and loading to the corresponding PLC: The file swinfo.txt -contains the corresponding information on the controller. → **Software information (swinfo.txt)** (→ p. [194](#))

### 7.1.5 Assign memory allocation – safety / standard PLC

24519

 When creating a project using the project template, the following memory layout is set by default:  
MemoryLayout\_2\_25s\_2\_25 = 2.25 MB safety PLC / 2.25 MB standard PLC



## WARNING

When changing the preset memory layout.

- > Risk of personal injuries and/or damage to property.
- > Loss of the conditions for safety operation.
- ▶ For safety operation of the unit, use the preset configuration 2 (= MemoryLayout\_2\_25s\_2\_25).
- ▶ When changing the memory layout, safety operation is not permissible.

More information: → **Memory allocation variants** (→ p. 41)

To allocate the memory layout to the PLCs:

### 1 Select memory layout

- ▶ In the device tree: Double-click on the symbol [CR7xxS (CR7xxS)]
- ▶ In the editor window: Select the [Memory Layout] tab.
- > The editor window shows the memory layout:

configuration	memory	IEC code Safety PLC	IEC code standard PLC	Memory layout
Configuration 1		1.0 Mbytes	3.5 Mbytes	MemoryLayout_1_00s_3_50
Configuration 2 (default)		2.25 Mbytes	2.25 Mbytes	MemoryLayout_2_25s_2_25
Configuration 3		3.5 Mbytes	1.0 Mbytes	MemoryLayout_3_50s_1_00

### 2 Set memory layout

- ▶ Highlight the required memory partition
- ▶ Click on the [Update device] button
- > Memory partitioning is adopted in CODESYS

### 3 Load memory partition into the device

- ▶ Click on the [Download configuration] button
- > The memory layout is downloaded to the device



When the memory layout is loaded to the device, the two IEC applications (standard application and safety application) will be deleted!

## 7.1.6 Assign inputs/outputs - safety / standard PLC

58720



When creating a project using the project template, all inputs and outputs are assigned to the standard PLC by default.

If an input/output is accessed that is not assigned to the respective PLC, the diagnostic message DIAG\_ACCESS appears at the corresponding function block.



The message only appears if there is not already an error, since the error message has a higher priority than the diagnostic message.

Access to the input/output is blocked by the system in all cases.



- ▶ Please ensure that the configurations of the inputs and outputs in the file CR7nnS\_iomapping.cfg and in CODESYS [[E/A-Zuweisung] are exactly identical! The settings must be identical!



## WARNING

Wrong assignment of the fail-safe inputs and outputs.

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function.
- ▶ It is mandatory that the project engineer verifies correct assignment of the safe inputs and outputs to the safety PLC by means of a function test.

To assign the I/Os to the PLCs, the following steps are required:

❗ Before the programming of the application may even begin:

### 1 Set the I/O assignment in the configuration file CR7nnS\_iomapping.cfg

In the sub-folder config, the device package includes the configuration file CR7xxS\_iomapping.cfg that is suitable for the device.

- ▶ Open the file CR7xxS\_iomapping.cfg in a text editor
- > The file has a section for each input and each output. It determines whether the input/output is assigned to the standard or the safety PLC:

Section / file entry	Description
[IN0000] Safe=FALSE	Assignment of the input IN0000 to the <b>standard PLC</b>
[IN0000] Safe=TRUE	Assignment of the input IN0000 to the <b>safety PLC</b>

- ▶ Define the assignment to a PLC of each input and output in accordance with the above table.
- ▶ Save the configuration file

### 2 Load the configuration file CR7nnS\_iomapping.cfg into the device using the update tool

- ▶ Execute the batch file Update.bat from the corresponding package.
- ❗ The version number of the package must be identical with the operating system version installed on the device!
- ▶ Follow the instructions on the screen.
- ▶ Open the menu item [Continue Update Process] with [u] > [ENTER].
- > The [ecomatController Update Menu] appears.
- ▶ Open the menu item [Write iomapping.cfg To Device] with [o] > [ENTER].
- > The file is transferred to the device into the folder cfg and renamed into iomapping.cfg. If there is already another file, it will be overwritten.

### 3 Select I/O assignment in CODESYS

- ▶ In the CODESYS device tree: Double-click on the symbol [CR7xxS (CR7xxS)]
- ▶ In the editor window: Select the [E/A-Zuweisung] tab.
- > Editor window shows the PLC assignment of the inputs and outputs (excerpt):

Section	Element	Parameter	Standard PLC	Safety PLC
System_Info	IP Settings	--	⊙	○

Section	Element	Parameter	Standard PLC	Safety PLC
Local_IO	Inputs	IN0000	⊙	○
		IN0001	⊙	○
		IN0002	⊙	○
		...	...	...
	Outputs	OUT0000	⊙	○
		OUT0001	⊙	○
		OUT0002	⊙	○
		...	...	...
	System_Outputs	VBB0_SW	⊙	○
		VBB1_SW	⊙	○
		...	...	...
		Supply_Switch	⊙	○
HMI	User_LEDs	User LED 0	⊙	○
		User LED 1	⊙	○
		User LED 2	⊙	○
		User LED 3	⊙	○

#### 4 Set the I/O assignment in CODESYS

- ▶ Highlight all I/Os in the column [StandardPLC] to allocate it to the **standard PLC**
- ▶ Highlight all I/Os in the column [SafetyPLC] to allocate it to the **Safety PLC**
- > The I/Os are allocated

### 7.1.7 Manage files

24828



When the transmission starts, the entire controller goes into the UPDATE mode.

In this mode, all outputs of the device will be switched off and both applications (standard and safety application ) will be stopped.

→ **Operating states** (→ p. [198](#))




For this action, you need to sign in as device user.

For more user convenience:

- ▶ Create a user in the project and sign the user in once
  - → **Create a user in the CODESYS project** (→ p. [77](#))
  - → **Sign the user in to the CODESYS project** (→ p. [78](#))

To transfer files between the PC and the device:

#### 1 Select file view

- ▶ In the device tree: Double-click on the symbol [CR7xxS (CR7xxS)]
- ▶ In the editor window: Select the [Files] tab.
- ▶ Click on the symbol  [Refresh]
- > The editor window shows the folder structure on the PC on the left and on the device on the right

#### 2 Transfer file from PC to device

- ▶ Highlight the file on the left
- ▶ Select device target directory on the right
- ▶ Start transfer using the [>>] button
- > The file is transferred to the device

#### 3 Transfer the file from the device to the PC

- ▶ Highlight the file on the right
- ▶ Select PC target directory on the left
- ▶ Start transfer using the [<<] button
- > The file is transferred to the PC

### 7.1.8 User-defined data

60505

The user can store any data in files (user files) in the directory data on the device. The files can be read and written from the standard application and from the safety application with the CODESYS library SysFile.



- ▶ Take suitable measures to secure the data against falsification, modification, unauthorised access and loss, e.g.: CRC32, plausibility checks, separation of safety data and non-safety data into different files, etc. The implementation of the measures is the responsibility of the project engineer.
- ▶ Note the following restrictions when accessing user-defined files:
  - Only open a maximum of 4 files simultaneously.
  - Only execute the functions of the CODESYS SysFile library in low priority tasks. The watchdog time must be long enough to execute the file operations.
  - Use a path length (file name incl. folder) with a maximum of 23 characters.
  - The following special characters are permitted in the file name: "\_", ".", "-".
  - Do not copy or delete the file /data/journal.jnl.
  - The maximum available usable memory for custom files is 768 KB (FAT32).
  - When transferring with tftp please note: The total file size of the transferred files must not exceed 768 KB, otherwise the file system will be damaged. In that case, a factory reset is required.
  - The SysFileCopy function supports only one copy operation at a time.
  - The SysFileCopy function copies only 256 bytes per call-up. The return value is ERR\_PENDING if further calls are necessary to copy the file completely. The return value ERR\_OK indicates that the file has been completely copied.

## File system - read/write performance

60506

The device achieves the following measured write performance:

Number of bytes	Standard PLC Min. time [ms]	Standard PLC Average time [ms]	Standard PLC Max. time [ms]	Safety PLC Min. time [ms]	Safety PLC Average time [ms]	Safety PLC Max. time [ms]
256	245	248	255	236	238	241
1024	230	346	379	243	350	388
8192	589	639	737	606	669	760
32768	1830	1904	1935	1854	1871	1899

The device achieves the following measured reading performance:

Number of bytes	Standard PLC Min. time [ms]	Standard PLC Average time [ms]	Standard PLC Max. time [ms]	Safety PLC Min. time [ms]	Safety PLC Average time [ms]	Safety PLC Max. time [ms]
256	13	14.4	17	13	14	16
1024	16	17	19	15	16.2	17
8192	26	27.8	30	26	27	28
32768	25	27.4	30	25	26.4	28



## 7.2 Configuring the standard PLC

### Content

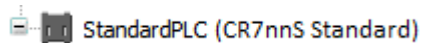
Configure task processing .....	86
Add function libraries to the application.....	87

24545



- ▶ Familiarise yourself with the following CODESYS functions!
  - Generic device editor
    - Online help > CODESYS Development System > Reference user interface > Objects > object 'device' and generic device editor

The Standard PLC of the CR7xxS is configured via the "generic device editor" of the CODESYS programming system. The programmer can access the device editor of the Standard PLC via the following node in the device tree:



To configure the device-internal Standard PLC of the CR7xxS:

- ▶ In the device tree: Double-click on [StandardPLC (CR7xxS Standard)]
- > The editor window shows the device editor of the device-internal Standard PLC.
- ▶ Configure Standard PLC.
- ▶ Save the project to apply changes.

25122



Add useful information to the standard and the safety application  
→ **Enter information about applications** (→ p. [78](#))

## 7.2.1 Configure task processing



- ▶ Familiarise yourself with the following CODESYS functions!
  - Task configuration:
    - Online help > CODESYS Development System > Program application > Task configuration

Parameters control the processing of the tasks. The user can set the parameters for each task:

- ▶ Create new task. (maximum 4 per PLC)
- ▶ Configure task properties:
  1. [Priority]: admissible = 0 (high) ... 3 (low)  
(select a priority for each task)
  2. [Type]: Cyclic
  3. [Interval]: Interval of the task call-ups in [ms]  
The interval time must be longer than the processing time of the task.
- ▶ Recommended: Activate watchdog: → **Configure IEC watchdog** (→ p. [93](#))  
The watchdog time must be shorter than the interval time.  
The watchdog time must be longer than the processing time of the task.
- ▶ Assign the sub-program with POU's to the newly created task.

Requirements for task interval time and watchdog time:

	minimum admissible	maximum admissible
Task interval time	4 ms	2000 ms
Watchdog time	2 ms	1998 ms

When the project is created, CODESYS automatically creates the following task using the project template:

Name	Description	Default settings
Task	Task for the processing of the main program [PLC_PRG (PRG)]	Priority: 1 Type: Cyclic Interval: t#10ms Watchdog enabled Time: t#8ms Sensitivity: 1



By default, CODESYS assigns the CAN communication to the task with the shortest interval time. If this is not wanted or if the CAN buses are operating at very high capacity:

- ▶ Configure task properties:
  1. [Priority]: high
  2. [Type]: Cyclic
  3. [Interval]: requested cycle time (=transmission interval)
- ▶ Assign sub-programs with the POU's for CAN communication to the CAN tasks.



- Please note both for the standard and the safety application:
- Multitasking the access to inputs and outputs with function blocks is only supported by the CR7xxS with restrictions.
  - Only create full and parameter setting access to an input from one task!  
For access from other tasks, only the MODE\_INPUT = MONITOR may be used.  
( → **MODE\_INPUT (ENUM)** (→ p. [286](#))
  - Only create full and parameter setting access to an output from one task!  
For access from other tasks, only the MODE\_OUTPUT = MONITOR may be used.  
( → **MODE\_OUTPUT (ENUM)** (→ p. [287](#)))



More information on task configuration:  
→ **Task configuration example** (→ p. [489](#))

## 7.2.2 Add function libraries to the application

The ifm package includes ifm function libraries for the programming of the device under CODESYS. The libraries are installed in CODESYS together with the ifm package. Users can add individual libraries to an application they need for the programming.



- Familiarise yourself with the following CODESYS functions!
- Library manager  
→ Online help > CODESYS Development System > Use libraries > Adding a library to the application

## 7.3 Configuring the safety PLC

### Content

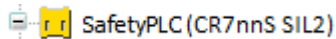
Configuration of the safety task processing .....	89
Add function libraries to the safety application .....	89

24546



- ▶ Familiarise yourself with the following CODESYS functions!
  - Generic device editor
    - Online help > CODESYS Development System > Reference user interface > Objects > object 'device' and generic device editor
  - Editor of the safety controller
    - Online help > add-ons > CODESYS Safety SIL2 > Device editor - safety

The fail-safe PLC of the CR7xxS can be configured via the device editor of the safety controller of the CODESYS programming system. The programmer can access the device editor of the fail-safe PLC via the following node in the device tree:



To configure the device-internal fail-safe PLC of the CR7xxS:

- ▶ In the device tree: Double-click on [SafetyPLC (CR7xxS SIL2)]
- > The editor window shows the device editor of the fail-safe PLC of the CR7xxS.
- ▶ Configure the fail-safe PLC as required.
- ▶ Save the project to apply changes.

25122



- Add useful information to the standard and the safety application
  - **Enter information about applications** (→ p. [78](#))

### 7.3.1 Configuration of the safety task processing

24790

The task configuration in the safety PLC is done in the same way as the task configuration in the standard PLC ( → **Configure task processing** ( → p. [86](#))), but the following is to be taken into consideration:

43525



#### WARNING

Incorrect configuration of the safety task processing.

- > Failure of the safety function.
- > The safety function will be executed too late.
- > Risk of personal injuries and/or damage to property.
- ▶ Please note for the safety application:
  - Configure the [Type] of the safety task as `cyclic`
  - "Safety-Task-[Interval]" must be longer than "cycle time"
  - Enable the watchdog for the safety task ( → **Configure IEC watchdog** ( → p. [93](#)))
  - "Watchdog [Time]" must be shorter than "Safety Task [Interval]"
  - "Watchdog [Time]" must be longer than "cycle time"
  - The following applies to Basic Level: [Sensitivity] must be 1
- ▶ Calculating the safety time
- ▶ Ensure that the safety time is sufficient to comply with the process safety time ( → **The process safety time** ( → p. [30](#)))

24863



- ▶ Please consider when setting up the safety function:

Changing the default settings affects the time-related behaviour of the diagnostics and the function!

→ **Time response** ( → p. [29](#))

25243



- ▶ Please note both for the standard and the safety application:
  - Multitasking the access to inputs and outputs with function blocks is only supported by the CR7xxS with restrictions.
  - Only create full and parameter setting access to an input from one task!  
For access from other tasks, only the `MODE_INPUT = MONITOR` may be used.  
( → **MODE\_INPUT (ENUM)** ( → p. [286](#)))
  - Only create full and parameter setting access to an input from one task!  
For access from other tasks, only the `MODE_OUTPUT = MONITOR` may be used.  
( → **MODE\_OUTPUT (ENUM)** ( → p. [287](#)))



More information on task configuration:

→ **Task configuration example** ( → p. [489](#))

### 7.3.2 Add function libraries to the safety application

25039

→ **Add function libraries to the application** ( → p. [87](#))

## 7.4 Use memory

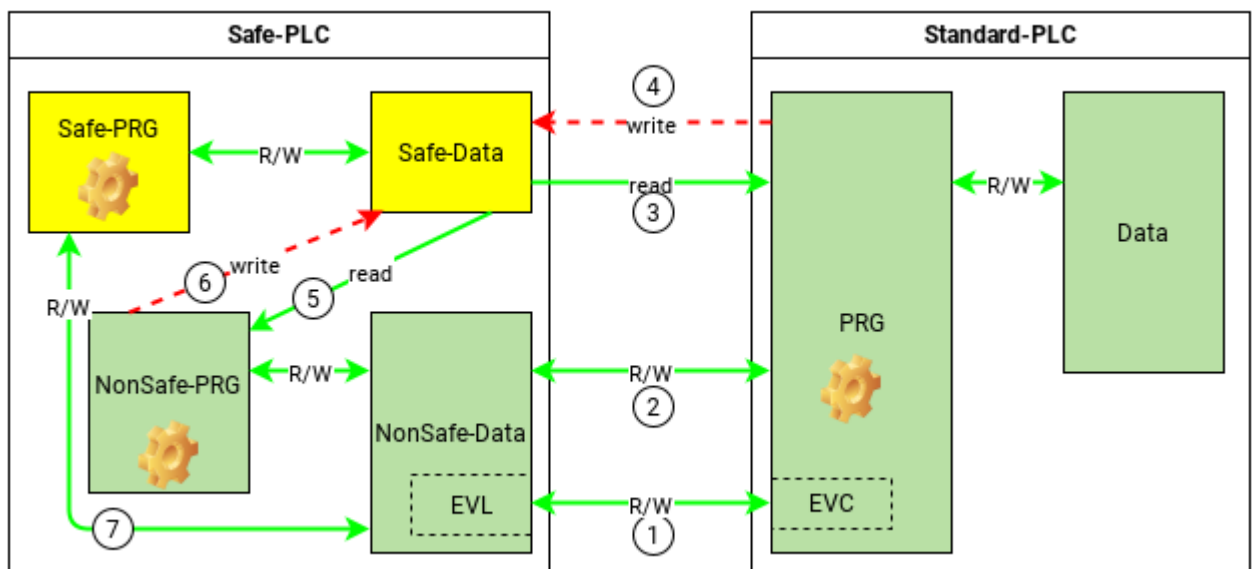
### Content

Memory protection .....	90
Data exchange between standard PLC and safety PLC .....	92
Stack .....	92

60507

### 7.4.1 Memory protection

60508



The IEC-RAM (Safe Data) of the safety PLC contains global variables, variables of safe PRGs and safety function blocks.

The IEC-RAM (Safe Data) of the safety PLC is protected against:

- Write access from the standard PLC
- Write access from non-safe PRGs of the Safety PLC
- Random errors via Error Correcting Code (ECC)

In case of unauthorised write access (arrows 4 and 6) from the non-safe PRG or the standard PLC, the accessing PLC goes into SERIOUS ERROR. The inputs and outputs of the accessing PLC go into the safe state.

Read access is technically possible throughout the system, but the following applies

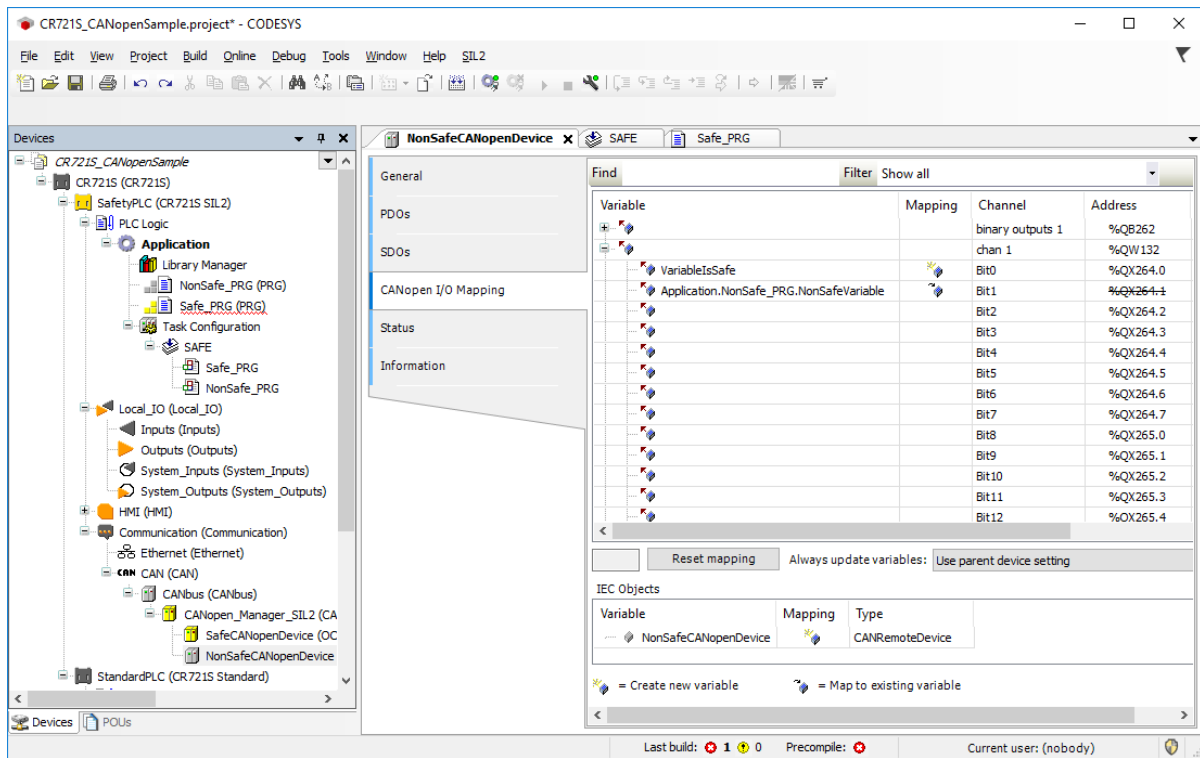
- Use EVC/EVL (arrow 1) for read and write access between safety and standard PLC. → **Data exchange between standard PLC and safety PLC** (→ p. 92)
- Use the NonSafe Data Area or EVL for data exchange between NonSafe PRG and Safe PRG (arrow 7).

#### Special behaviour of IO-mapping variables of the safety PLC:

If new IO-mapping variables are created, CODESYS automatically assigns them to the Safe Data memory area (mapping symbol with a star).

If write access is to be made to an IO mapping variable from a non-safe PRG, the IO mapping must be mapped to a variable already existing in the standard PRG (mapping symbol with arrow).

Typical example application: Access to a non-safe CANopen device on a CANopen Safety Manager of the Safety PLC.



## 7.4.2 Data exchange between standard PLC and safety PLC

54329



- ▶ Familiarise yourself with the following CODESYS functions!
  - EVC
    - Online help > [Add-ons] > [CODESYS Safety SIL 2] > [Compound Safety PLC] > [EVC, Exchange Variable Connection]
  - EVL
    - Online help > [Add-ons] > [CODESYS Safety SIL 2] > [Compound Safety PLC] > [EVL, Exchange Variable List]
  - Handling
    - Online help > [Add-ons] > [CODESYS Safety SIL 2] > [Compound Safety PLC] > [Handling of EVC/EVL]

The data between the standard PLC and the safety PLC is exchanged via EVC (Exchange Variable Connection) and EVL (Exchange Variable List):

PLC	Mechanism	Description
Standard PLC	EVC	The variables that are to be used from the safety application in the standard application can be selected in an editor.
Safety PLC	EVL	Special type of the GVL (Global Variable List) to declare variables that are used to exchange non safe data between a safety application and a standard application.  All variables of the EVL are stored in the non safe memory area. This enables write access from the standard application.

25124



As a general rule, the variable values between the Standard PLC and the fail-safe PLC are not transferred as fail-safe data!

- ▶ Fail-safe variable values do not necessarily imply fail-safe functions!
- ▶ Always transfer safe information created in the fail-safe area to the standard area unchanged. The signal can then be further processed in the standard area (e.g. negation).

## 7.4.3 Stack

60509

For each IEC task, 2.5 KB of stack are available to the IEC application.

Variables of IEC functions (inputs, outputs, local variables) are stored on the stack.

Especially nested calls of functions as well as functions with strings, arrays and structures occupy a lot of stack.

The stack consumption is monitored. If the maximum available stack is exceeded, the machine goes into FATAL\_ERROR.

Programming measures for stack optimisation:

- ▶ Divide nested function calls into several individual calls.
- ▶ Save local variables as global variables.
- ▶ Convert functions into function blocks.



## 7.5 Configure IEC watchdog

24862



- Familiarise yourself with the following CODESYS functions!

- Watchdog:  
→ Online help > CODESYS Development System > Programming of Applications > Task Configuration > Creating a Task Configuration > Tab 'Configuration'
- Task configuration:  
→ Online help > CODESYS Development System > Programming of Applications > Task Configuration



Please note for the safety application:

- Enable the watchdog for the safety task!

To configure the IEC watchdog of a task:

- Open the task configuration  
Standard task: → **Configure task processing** (→ p. [86](#))  
Safety task: → **Configuration of the safety task processing** (→ p. [89](#))
- Enable the watchdog with the [Enable] option field
- Enter watchdog [Time]
- Set [Sensitivity]
- > Watchdog is configured



The watchdog time must be shorter than the interval time.

The watchdog time must be longer than the processing time of the task.

## 7.6 Configure interfaces

### Content

Configure serial interface.....	94
Configure Ethernet interface.....	94
Configuring CAN interfaces.....	96
Interface configuration file comconf.cfg.....	104

39488

### 7.6.1 Configure serial interface

39473

The CODESYS service communication via RS232 only works with the preset baud rate.

For other purposes, the device supports the following baud rates:

9 600 baud  
 19 200 baud  
 38 400 baud  
 57 600 baud  
 115 200 baud (preset)

Setting the interface: → **Interface configuration file comconf.cfg** (→ p. [104](#))

### 7.6.2 Configure Ethernet interface

24994

Setting the interface: → **Interface configuration file comconf.cfg** (→ p. [104](#))

Factory setting:

IP address = 192.168.82.247  
 Subnet mask = 255.255.255.0  
 Gateway address = 192.168.82.21  
 UDP-Port = 12345

### NOTICE!

If the device is operated in an unprotected network environment.

- > Unauthorised data access (read or write) is possible.
- > Unauthorised manipulation of the device functions is possible.
- ▶ Check and restrict access options to the device:
  - Restrict access to authorised persons.
  - Do not connect the device to open networks or the internet.
  - If access from the internet should be necessary, it is mandatory to choose a secure method to connect the device (e.g. VPN).

24866



- ▶ Receiving too many data packages may lead to data loss!



### Setting the IP parameter of the Ethernet interface

39623

In order to update the runtime system of the CR7xxS via a network, the device must be connected to the corresponding network. For the configuration of the Ethernet interface, the following options are available:

- **Manual**                      The user defines the parameters of the Ethernet interface manually:  
IP address,  
Subnet mask,  
gateway address  
    ► Observe the **Address assignment in Ethernet networks** (→ p. [96](#)) in  
    Ethernet networks!
- **Automatic**                The interface parameters are set via the Dynamic Host Configuration  
Protocol (DHCP).  
(the development of this function is still in progress)

Setting the interface: → **Interface configuration file comconf.cfg** (→ p. [104](#))

## Address assignment in Ethernet networks

39571



In the Ethernet network every IP address **MUST** be unique.

The following IP addresses are reserved for network-internal purposes and are therefore not allowed as an address for participants: nnn.nnn.nnn.0 | nnn.nnn.nnn.255.

Only network participants whose subnet mask is identical and whose IP addresses are identical with respect to the subnet mask can communicate with each other.

### Rule:

If part of the subnet mask = 255, the corresponding IP address parts must be identical.

If part of the subnet mask = 0, the corresponding IP address parts must be different.

If the subnet mask = 255.255.255.0, 254 participants communicating with each other are possible in the network.

If the subnet mask = 255.255.0.0, 256x254 = 65 024 participants communicating with each other are possible in the network.

In the same physical network different subnet masks of the participants are allowed. They form different groups of participants which cannot communicate with groups of participants having other subnet masks.



In case of doubt or problems please contact your system administrator.

### Examples:

Participant A IP address	Participant A Subnet mask	Participant B IP address	Participant B Subnet mask	Communication of participants possible?
192.168.82.247	255.255.255.0	192.168.82.10	255.255.255.0	Yes, 254 participants possible
192.168.82. <b>247</b>	255.255.255.0	192.168.82. <b>247</b>	255.255.255.0	No (same IP address)
192.168.82.247	255.255. <b>255</b> .0	192.168.82.10	255.255. <b>0</b> .0	No (different subnet mask)
192.168. <b>82</b> .247	255.255.255.0	192.168. <b>116</b> .10	255.255.255.0	No (different IP address range: 82 vs. 116)
192.168.222.213	255.255.0.0	192.168.222.123	255.255.0.0	Yes, 65 024 participants possible
192.168.111.213	255.255.0.0	192.168.222.123	255.255.0.0	Yes, 65 024 participants possible
192.168.82.247	255.255.255.0	192.168.82. <b>0</b>	255.255.255.0	No; the whole network is disturbed because the IP address xxx.xxx.xxx.0 is not allowed

## 7.6.3 Configuring CAN interfaces

24708

The CAN interfaces are configurable as follows:

- via system configuration:
  - CANopen
  - SAE J1939
- via function block:
  - RAW-CAN

Under Vendor = 3S, you will find, among others, the following entries:

- CIA CANopen
  - CIA CANopenManager
    - + - CANopen\_Manager
    - + - CANopen\_Manager\_SIL2
  - CIA Local Device
    - + - CANopen Device
    - + - CANopen Device SIL2
- SAE J1939
  - SAE J1939 Manager
    - + - J1939\_Manager

The following protocols are only available for safety operation:

- CANopen\_Manager\_SIL2
- CANopen\_Device\_SIL2



- ▶ When configuring CANopen and SAE J1939 via system configuration, please note the following:
  1. In the Manager component, select the task suitable for CAN communication in [IO Mapping] under [Bus cycle options].  
Otherwise, CODESYS will attach the communication stack to the task with the shortest task interval. This might lead to an or unexpected time response of the CAN bus.
  2. Protocol messages (e.g. SYNC, PDO, PGN/SPN, etc.) can only be processed at the interval of the set task.
  3. The task interval should not exceed half of the bus interval (SYNC, repetition time, etc.).
  4. In the settings of the CANopen manager, the validity of the time settings can be verified via [Konfiguration prüfen und korrigieren].

## via system configuration: CANopen Manager

39620

In the CODESYS device tree, you will find the following entry under each PLC:  
[Communication] > [CAN]



Configure each interface only at ONE position!

- **Attach CAN bus**
  - ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN].
  - ▶ Select [Add Device...].
  - > Window [Add Device] appears.
  - ▶ Select [Vendor:] [ifm electronic].
  - ▶ In the list below: Select [ifmCANbus].
  - ▶ Confirm the selection with [Add Device].
  - > Close the window [Add Device] with the [Close] button.
- **Assign CAN interface**
  - ▶ In the CODESYS device tree: Double-click on [Communication] > [CAN] > [ifmCANBus].

- ▶ Tab [General] > [General] > [Network]:  
assign this setting with ▲/▼ to a CAN interface.  
permissible = 0...3
- ▶ Select the required value for baud rate [Baudrate (bit/s)] from the list field.
- **Attach CANopen manager**
  - ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN] > [ifmCANBus].
  - ▶ Select [Add Device...].
  - > Window [Add Device] appears.
  - ▶ Select [Vendor:]: [<All vendors>].
  - ▶ In the list below: Select [Fieldbusses] > [CiA CANopen] > [CiA CANopenManager] > [CANopenManager].
  - ▶ Confirm the selection with [Add Device].
  - ▶ Close the window [Add Device] with the [Close] button.
- **Set CANopen manager parameters**
  - ▶ In the CODESYS device tree: Double-click on [Communication] > [CAN] > [CiA CANopenManager] > [CANopenManager].
  - ▶ Tab [General] > [General] > [Node ID]:  
assign a node ID to this interface using ▲/▼.  
permissible = 1...127
  - ▶ Select the further parameters according to the requirements, e.g.:
    - Configure the heartbeat protocol in the section [Nodeguarding]:
      - ▶ Click on the field of options in order to activate [Heartbeat Producing]
      - ▶ Set the parameters [Node ID] and [Producer Time (ms)]
    - Configure the sync protocol in the section [Sync]:
      - ▶ Click on the field of options to activate [Enable Sync Producing], if necessary.
      - ▶ Set the parameters [COB-ID (Hex)], [Cycle Period (µs)] and [Window Length (µs)]
    - In the section [Time]: The time protocol is not supported.
  - > With the menu [File] > [Save Project], the values become valid.



The sync protocol triggers the receiving/sending of data of the CANopen devices (input: SDO 16#1800/ output: SDO 16#1400).

- **Attach CANopen remote device**
  - ▶ In the CODESYS device tree: Right-click [Communication] > [CAN] > [ifmCANBus] > [CANopen\_Manager].
  - ▶ Select [Add Device...].
  - > The window [Add Device] appears.
  - ▶ In the area [Vendor:], select [<All vendors>].
  - ▶ In the list below: Under [Fieldbusses] > [CiA CANopen] > [Remote Device], select a remote device.
  - ▶ Confirm selection with [Add Device].
  - ▶ Close the window [Add Device] with the button [Close].
- **Configuring a CANopen remote device**
  - ▶ In the CODESYS device tree: [Communication] > [CAN] > [ifmCANBus] > [CANopen\_Manager] > double-click remote device.

- ▶ Select/set the parameters as required



When the option field [Reset Node] is activated: Each time the controller is switched on or the program is downloaded, the settings are reset to factory settings. Thereby, user settings can be overwritten. The type and the scope of reset settings depend on the device.

- ▶ With the menu [File] > [Save Project] the values become valid.

## via system configuration: CANopen device

39619

In the CODESYS device tree, you will find the following entry under each PLC:  
[Communication] > [CAN]  
These entries are equivalent.

- **Attach CAN bus**

- ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN].
- ▶ Select [Add Device...].
- > Window [Add Device] appears.
- ▶ Select [Vendor:] [ifm electronic].
- ▶ In the list below: Select [ifmCANbus].
- ▶ Confirm the selection with [Add device].
- ▶ Close the window [Add device] with the [Close] button.

- **Assign CAN interface**

- ▶ In the CODESYS device tree: Double-click on [Communication] > [CAN] > [ifmCANBus].
- ▶ Tab [General] > [General] > [Network]:  
assign this setting with ▲/▼ to a CAN interface.  
permissible = 0...3
- ▶ Select the required value for baud rate [Baudrate (bit/s)] from the list field.

- **Attach CANopen device**

- ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN] > [ifmCANBus].
- ▶ [Add Device...] Select .
- > Window [Add Device] appears.
- ▶ Select [Vendor:] <All vendors>.
- ▶ In the list below: Select [Fieldbusses] > [CiA CANopen] > [Local Device] > [CANopenDevice].
- ▶ Confirm the selection with [Add Device].
- ▶ Close the window [Add Device] with the [Close] button.

- **Set CANopen device parameters**

- ▶ In the CODESYS device tree: Double-click on [Communication] > [CAN] > [CANopenDevice].
- ▶ Tab [General] > [General] > [Node ID]:  
assign a node ID to this interface using ▲/▼.  
permissible = 1...127
- ▶ Select the further parameters according to the requirements e.g. [Heartbeat Producing].
- > With the menu [File] > [Save Project], the values become valid.



In the CANopen Device, the correct baud rate and the node ID must be set, so that the CANopen master will recognise the device.

## via system configuration: CANopen manager SIL2

In the CODESYS device tree, you will find the following entry under each PLC:

[Communication] > [CAN]



Configure each interface only at ONE position!

- **Attach CAN bus**
  - ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN].
  - ▶ Select [Add Device...]
  - > Window [Add Device] appears.
  - ▶ In the area [Device]: Make: Select [ifm electronic].
  - ▶ In the list below: Select [ifmCANbus].
  - ▶ Confirm the selection with [Add Device].
  - ▶ Close the window [Add Device] with the [Close] button.
- **Assigning the CAN interface**
  - ▶ In the CODESYS device tree: Double-click on [Communication] > [CAN] > [ifmCANBus].
  - ▶ Tab [General] > [General] > [Network]:  
assign this setting with ▲/▼ to a CAN interface.  
permissible = 0...3
  - ▶ Select the required value for [Baudrate (bit/s)] from the list field.
- **Attach CANopen manager SIL2**
  - ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN] > [ifmCANBus].
  - ▶ Select [Add Device...]
  - > Window [Add Device] appears.
  - ▶ In the area [Device]: Select [Vendor:] [<All Vendors>].
  - ▶ In the list below: Select [Fieldbusses] > [CiA CANopen] > [CiA CANopenManager] > [CANopen\_Manager\_SIL2].
  - ▶ Confirm the selection with [Add Device].
  - ▶ Close the window [Add Device] with the [Close] button.
- **Configure CANopen manager SIL2**
  - ▶ In the CODESYS device tree: Double-click [Communication] > [CAN] > [ifmCANBus] > [CANopen\_Manager\_SIL2].
  - ▶ Tab [General] > [General] > [Node ID]:  
assign a node ID to this interface using ▲/▼.  
permissible = 1...127
  - ▶ Select the further parameters according to the requirements, e.g.:
    - Configure the heartbeat protocol in the section [Guarding]:
      - ▶ Click on the field of options in order to activate [Heartbeat-Producing]
      - ▶ Set the parameters [Node ID] and [Producer Time (ms)]
    - Configure the sync protocol in the section [Sync]:
      - ▶ Click on the field of options to activate [Enable Sync-Producing], if necessary.
      - ▶ Set the parameters [COB-ID (Hex)], [Cycle Period (µs)] and [Window Length (µs)]
    - In the section [Time]: The time protocol is not supported.



- > With the menu [File] > [Save Project], the values become valid.



The sync protocol triggers the receiving/sending of data of the CANopen devices (input: SDO 16#1800/ output: SDO 16#1400).

- **Attaching a CANopen remote device**
  - ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN] > [ifmCANBus] > [CANopen\_Manager\_SIL2].
  - ▶ Select [Add Device...]
  - > Window [Add Device] appears.
  - ▶ In the area [Device]: Select [Vendor:] [<All Vendors>].
  - ▶ In the list below: Select a remote device under [Fieldbusses] > [CiA CANopen] > [CiA Remote Device].
  - ▶ Confirm the selection with [Add Device].
  - ▶ Close the window [Add Device] with the [Close] button.
- **Configuring a CANopen remote device**
  - ▶ In the CODESYS device tree: [Communication] > [CAN] > [ifmCANBus] > [CANopen\_Manager\_SIL2] > double-click remote device.
  - ▶ Select/set the parameters as required



When the option field [Reset Node] is activated: Each time the controller is switched on or the program is downloaded, the settings are reset to factory settings. Thereby, user settings can be overwritten. The type and the scope of reset settings depend on the device.

- > With the menu [File] > [Save Project], the values become valid.

## via system configuration: CANopen-Device SIL2

24865

In the CODESYS device tree, you will find the following entry under each PLC:  
[Communication] > [CAN]

These entries are equivalent.

- **Attach CAN bus**
  - ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN].
  - ▶ Select [Add Device...]
  - > Window [Add Device] appears.
  - ▶ In the area [Device]: Make: Select [ifm electronic].
  - ▶ In the list below: Select [ifmCANbus].
  - ▶ Confirm the selection with [Add Device].
  - ▶ Close the window [Add Device] with the [Close] button.
- **Assign CAN interface**
  - ▶ In the CODESYS device tree: Double-click on [Communication] > [CAN] > [ifmCANBus].
  - ▶ Tab [General] > [General] > [Network]:  
assign this setting with ▲/▼ to a CAN interface.  
permissible = 0...3
  - ▶ Select the required value for baud rate [Baudrate (bit/s)] from the list field.

- **Attach CANopen device SIL2**

- ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN] > [ifmCANBus].
- ▶ Select [Add Device...]
- > Window [Add Device] appears.
- ▶ In the area [Device]: Make: select <All Vendors>.
- ▶ In the list below: Select [Fieldbusses] > [CiA CANopen] > [Local Device] > [CANopenDevice SIL2].
- ▶ Confirm the selection with [Add Device].
- ▶ Close the window [Add Device] with the [Close] button.

- **CANopen-Device SIL2 parametrieren**

- ▶ In the CODESYS device tree: Double-click on [Communication] > [CAN] > [Local Device] > [CANopenDevice].
- ▶ Tab [General] > [General] > [Node ID]:  
assign a node ID to this interface using ▲/▼.  
permissible = 1...127
- ▶ Select the further parameters according to the requirements, e.g.: [Heartbeat-Producing]
- > With the menu [File] > [Save Project], the values become valid.



In the CANopen Device SIL, the correct baud rate and the node ID must be set, so that the CANopen master will recognise the device.

## via system configuration: J1939 manager

39615

In the CODESYS device tree, you will find the following entry under each PLC:  
[Communication] > [CAN]  
These entries are equivalent.

- **Attach CAN bus**

- ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN].
- ▶ Select [Add Device...].
- > Window [Add Device] appears.
- ▶ Select [Vendor:] [ifm electronic].
- ▶ In the list below: Select [ifmCANbus].
- ▶ Confirm the selection with [Add Device].
- > Close the window [Add Device] with the [Close] button.

- **Assign CAN interface**

- ▶ In the CODESYS device tree: Double-click on [Communication] > [CAN] > [ifmCANBus].
- ▶ Tab [General] > [General] > [Network]:  
assign this setting with ▲/▼ to a CAN interface.  
permissible = 0...3
- ▶ Select the required value for baud rate [Baudrate (bit/s)] from the list field.

- **Attach J1939 manager**

- ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN] > [ifmCANBus].
- ▶ Select [Add Device...].
- > Window [Add Device] appears.
- ▶ Select [Vendor:] <All Vendors>.

- ▶ In the list below: Select [Fieldbusses] > [SAE J1939] > [J1939 Manager] > [J1939\_Manager].
- ▶ Confirm the selection with [Add Device].
- ▶ Close the window [Add Device] with the [Close] button.

- **Set J1939 manager parameters**

- ▶ In the CODESYS device tree: Double-click [Communication] > [CAN] > [J1939\_Manager].
- ▶ Tab [General] > [Database] > [Database]:  
select the list from the required database.  
default = J1939Default



Users can use their own databases.

They must be at the following storage location:  
C:\ProgramData\CODESYS\J1939 Databases

The directory ProgramData is hidden by default.

- > With the menu [File] > [Save Project], the values become valid.

- **Attach J1939-ECU**

- ▶ In the CODESYS device tree: Right mouse click on [Communication] > [CAN] > [ifmCANBus] > [J1939\_Manager].
- ▶ Select [Add Device...].
- > Window [Add Device] appears.
- ▶ In the area [Device]: [Vendor:] select <All vendors>.
- ▶ In the list below: Select [Fieldbusses] > [J1939] > [J1939\_ECU] > [J1939\_ECU].
- ▶ Confirm the selection with [Add Device].
- ▶ Close the window [Add Device] with the [Close] button.

- **Set J1939-ECU parameters**

- ▶ In the CODESYS device tree: Double-click on [Communication] > [CAN] > [J1939\_Manager] > [J1939\_ECU].
- ▶ Make the following settings in the tab [General] in the section [General] according to the specific application:

user case	[Local Device]		Significance [Preferred Address]
<ul style="list-style-type: none"> <li>▪ Receiving broadcast data of the ECU</li> <li>▪ No transmission</li> </ul>	<input type="checkbox"/>	deactivated	Address of the ECU from which the data is to be received
<ul style="list-style-type: none"> <li>▪ Sending data (broadcast and P2P)</li> <li>▪ Receiving P2P data</li> </ul>	<input checked="" type="checkbox"/>	activated	Address of the ifm controller

- ▶ Add parameter groups in the tab [TX-Signals] by clicking on [Add PG].
- > The settings become valid with menu [File] > [Save Project].

## via function block: RAW-CAN

39614

The **Library ifmRawCAN.library** (→ p. [216](#)) features several function blocks for this application.

### 7.6.4 Interface configuration file comconf.cfg

25108

The file directory /com of the device contains the file comconf.cfg.

To change the configuration data of the following interfaces, this file must be written into the device with the corresponding changes:

- serial interface
- Ethernet interface
- CAN interfaces

Factory setting of the content:

```
[ETHERNET0]
IPv4Address=192.168.82.247
IPv4SubnetMask=255.255.255.0
IPv4Gateway=192.168.82.21

[CAN0]
Baud rate=250000
NodeId=127

[CAN1]
Baud rate=250000
NodeId=126

[CAN2]
Baud rate=250000
NodeId=125

[CAN3]
Baud rate=250000
NodeId=124

[COM0]
Baud rate=115200
Bits=8
Parity=0
Stop=1
```

- ▶ To start the (disabled) device with these default settings:  
(the file comconf.cfg is not taken into consideration)  
TRUE on connection RESET-COM (pin 72) simultaneously with POWER-ON  
After the start-up: FALSE on RESET-COM
- ▶ To start the (disabled) device with the content of the (new) file comconf.cfg:  
FALSE on connection RESET-COM (Pin 72) simultaneously with POWER-ON

Evaluation in the application with the variable COM\_RESET\_I → **Access inputs** (→ p. [118](#)).

## 7.7 Configure inputs and outputs

### Content

via function block .....	105
via safety function block .....	106
via system configuration .....	106

25009

The inputs and outputs can be configured as follows:

- via function block:
  - inputs
  - outputs
  - User LEDs
- via system configuration (via I/O image):
  - System inputs
  - System outputs

Further information on the use of inputs/outputs:

→ **Options to access the input data and output data** (→ p. [112](#))

For standard PLC:

→ **Access inputs** (→ p. [118](#))

→ **Access outputs** (→ p. [119](#))

For safety PLC:

→ **Access to safety input data and safety output data** (→ p. [136](#))

→ **1-channel safety concept for inputs** (→ p. [140](#))

→ **2-channel safety concept for inputs** (→ p. [150](#))

→ **Safety concept of the outputs** (→ p. [163](#))

### 7.7.1 via function block

24704

This method is used to configure inputs, outputs and the user LEDs during the processing time of the application.

For this, the operating mode of the inputs / outputs is set as follows.

Examples:

FB	FB input
<b>Input</b> (→ p. <a href="#">270</a> )	eMode
<b>Output</b> (→ p. <a href="#">273</a> )	eMode
<b>OutputGroup</b> (→ p. <a href="#">299</a> )	eMode

If the eMode at the function block is changed:

- Pending diagnostic messages will be reset
- Measured values will be reset to 0 / FALSE
- Diagnostic settings will be set to standard values

## 7.7.2 via safety function block

24703



► Please note additional information:

- → **Programming of the safety application** (→ p. [127](#))
- → **ifm function libraries** (→ p. [207](#))

for safety programming in the basic and extended level:

configure and process inputs and outputs that are assigned to the safety PLC with special safety function blocks.

For this, the operating mode of the inputs / outputs is set as follows.

Examples:

FB	FB input
<b>SF_Input</b> (→ p. <a href="#">322</a> )	eMode
<b>SF_Output</b> (→ p. <a href="#">357</a> )	eMode
<b>SF_OutputGroup</b> (→ p. <a href="#">360</a> )	eMode

## 7.7.3 via system configuration

24705

This method (via I/O image) is used to configure system inputs and system outputs.



Note: Only the I/Os assigned to the corresponding PLC can be configured!



The procedures to configure inputs and outputs via system configuration are identical for safety and standard PLCs!

Procedure using the example of the filter setting of a system input:

- In the CODESYS device tree: Extend required PLC > Element [Local\_IO]
- Double-click [System\_Inputs]
- Click on [Parameters] tab
- > The parameter view of the system inputs appears
- Select the system input in the list
- Double-click in the [Value] column of the [Filter] parameter
- Click on arrow symbol
- > A list with possible filter modes appears
- Click on filter mode
- > The filter for the system input is set
- If necessary, set further parameters as described, e.g. Value etc.
- Save the project to apply changes.

## 8 Programming of the standard application

### Content

Error in the IEC application.....	107
Objects of a standard PLC application .....	108
Creating a standard PLC application.....	109
Using ifm function libraries .....	116
Use IO mapping.....	118
Use RawCAN (CAN Layer 2) .....	122
Use CANopen.....	123
Use SAE J1939 .....	124
Using Ethernet.....	125

24524

### 8.1 Error in the IEC application

25159

The operating system (ifmOS) treats the following errors from the IEC application and leads the causing PLC into error class B. ( → **Error classes** (→ p. [462](#))



To improve the availability of the application, the project engineer should avoid the above-mentioned errors and avoid them e.g. by means of coding guidelines, review, test, etc.

Error in the application	Error class
Division by zero for Integer data type	B
Division by zero for Float data type	B
Memory access violation (writing)	B
Zero pointer exception	B

The operating system (ifmOS) has no error handling for the following arithmetic errors.

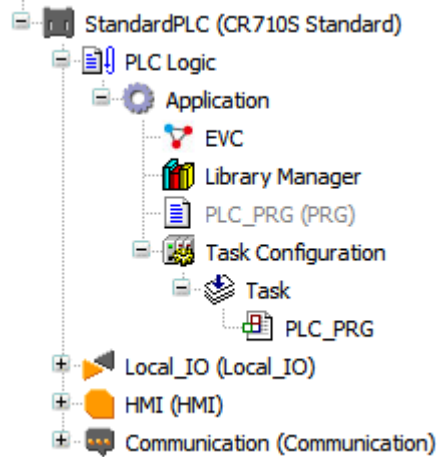


The project engineer must exclude the following errors by avoiding systematic errors and e.g. coding guidelines, review, test, etc. or allow the error calculations explicitly (e.g. subtraction of 2 values to get one absolute value).

Error in the application	Error class
Overflow (data types Integer and Float)	No error handling!
Underflow (data types Integer and Float)	No error handling!
Float Inexact Value	No error handling!

## 8.2 Objects of a standard PLC application

All objects of a standard PLC application are listed as sub-elements of the node [StandardPLC (CR7xxS Standard)] > [PLC Logic] > [Application] in the device tree. In the basic configuration, a standard PLC application includes the following objects:



[Application]	Container for objects of a PLC application
[EVC]	Exchange Variable Connection. Provides the standard PLC application with variables of the safety PLC application via a shared memory area. → <b>Data exchange between standard PLC and safety PLC</b> (→ p. <a href="#">92</a> )
[Library Manager]	Provides access to standard and device-specific function libraries: → <b>Using ifm function libraries</b> (→ p. <a href="#">116</a> )
[PLC_PRG(PRG)]	Provides access to the editor of the standard PLC application → <b>Creating a standard PLC application</b> (→ p. <a href="#">109</a> )
[Task Configuration]	Provides access to the settings of the task processing: → <b>Configure task processing</b> (→ p. <a href="#">86</a> )

If necessary, the user can add further objects to the standard PLC application.



## 8.3 Creating a standard PLC application

### Content

Supported programming languages .....	109
Data types.....	109
Supported variable types.....	110
Options to access the input data and output data .....	112
Default behaviour of the inputs and outputs in case of an error.....	113
Standard diagnostic limit values of inputs .....	113
Standard diagnostic limit values of outputs .....	114
Call sequence for diagnostic function blocks for inputs and outputs .....	114

24526



- ▶ Familiarise yourself with the following CODESYS functions!
  - → Online help > CODESYS Development System > Programming Applications

CODESYS automatically generates the function block PLC\_PRG (PRG) during project creation. The function block is processed cyclically. Other programs are called in this function block.

To create a PLC application:

- ▶ In the device tree: Double-click on [Application] > [PLC\_PRG (PRG)]
- > Editor window shows input mask of the selected programming language.
- ▶ Enter program code.

### 8.3.1 Supported programming languages

58723

The following programming languages according to IEC 61131 may be used in standard program blocks:

- Function block diagram FUP/FBD
- Ladder diagram KOP/LD
- Structured text ST
- Sequential function chart AS/SFC
- Continuous function chart CFC

### 8.3.2 Data types

25147



- ▶ Familiarise yourself with the following CODESYS functions!
  - Data types
    - Online help > CODESYS Development System > Reference, Programming > Data Types

The data types described in the CODESYS online help are available for:

- the standard application
- the safety application in the system level

### 8.3.3 Supported variable types

#### Content

Non-volatile data.....	110
Function description of non-volatile data.....	111

24746



► Familiarise yourself with the following CODESYS functions!

- Local variables  
→ Online help > CODESYS Development System > Reference, Programming > Variable Types and special Variables > Local Variables – VAR
- Global variable list  
→ Online help > CODESYS Development System > Reference, Programming > Variable Types and special Variables > Global Variables - VAR\_GLOBAL
- network variables UDP  
→ Online help > CODESYS Development System > Working with Control Networks > Network Variables
- Non-volatile variables - RETAIN PERSISTENT  
→ Online help > CODESYS Development System > Reference, Programming > Variable Types and special Variables > Remanent Variables - RETAIN, PERSISTENT

The device supports the following variable types:

Variable type	Declaration	Scope of validity	Application	Memory behaviour
local	In the declaration part of the POU	Applies only to the POU in which it has been declared	Standard / Safe	volatile
locally PERSISTENT				non volatile for: - reset warm/cold - download
global	In the global variable list (GVL)	Applies to all POUs of the project	Standard / Safe	Volatile
global PERSISTENT				non volatile for: - reset warm/cold - download
Network	In network variable lists	Values are available to all projects in the whole network if the variable is contained in their network variables lists. Only UDP network variables are supported.	Standard	volatile



For performance reasons, do not overuse 64 bit variables!

#### Non-volatile data

60510

Different types of non-volatile data are supported:

- Retain variables / retain data (VAR\_PERSISTENT)
- Memory bytes (%MB, %MW, etc.)

Variables declared as VAR\_PERSISTENT generate non-volatile data.

Memory bytes are pre-defined non-volatile memory areas.

In the following, both types will be referred to as non-volatile data.

Non-volatile data includes the values that are stored in it during the following processes:

- Switching the device on/off
- Online reset (warm/cold) of the device
- Loading the PLC (download)

Typical applications for non-volatile variables are, for example:

- Operating hours that are counted up and retained while the machine is in operation,
- Position values of incremental encoders,
- Target values entered in the display unit,
- Machine parameters,

i.e. all variables whose values must not get lost when the device is switched off.

## Function description of non-volatile data

60511

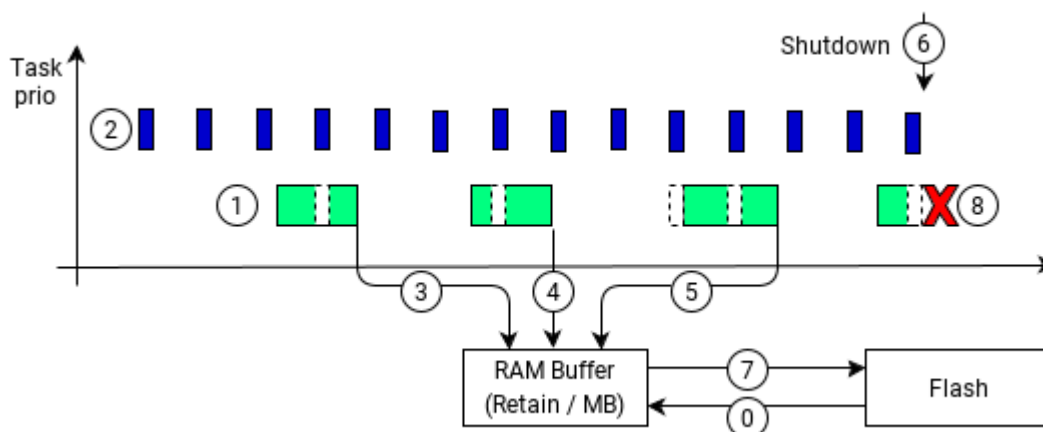
After power-on, the non-volatile data is written from the FLASH memory to the RAM intermediate buffer (0).

At the end of the cycle of the low-priority task (1), non-volatile data is written into an intermediate RAM buffer (3, 4, 5).

The execution of the lower priority task (1) is not interrupted by higher priority tasks (2) at the end of the cycle. This ensures the consistency of the non-volatile data.

In the SHUTDOWN state (6), the non-volatile data is written from the volatile RAM memory to the non-volatile flash memory (7).

Only the data of the last completely executed task will be stored in the RAM intermediate buffer (5). The data of an unfinished task (8) will not be stored.





## WARNING

Data loss in case of FATAL\_ERROR

- > Any non-volatile data in the RAM intermediate buffer that has changed since the application was started will be lost.
- > Personal injury and/or damage to property is possible.
- ▶ Measures to be taken depending on the application according to → **Non-volatile safety-related data (static)** (→ p. [135](#)) or → **Non-volatile safety-related data (dynamic)** (→ p. [136](#)) must be observed.



After occurrence of a FATAL\_ERROR followed by a Power-On reset of the controller, the data stored in the FLASH memory during the last SHUTDOWN will be available.



Please note the following information:

- → **Switch on/off via main switch** (→ p. [43](#))
- → **Switch on/off via ignition lock (clamp 15)** (→ p. [43](#))
- → **FB SupplySwitch** (→ p. [278](#))
- → **Operating states** (→ p. [198](#))
- → **Error classes** (→ p. [462](#))

### 8.3.4 Options to access the input data and output data

58724

In a CODESYS project, each input and output has a physical address according to the IEC standard (e.g. %IW5). CODESYS offers the following options to access this address from a Standard PLC application and thereby to access the input and output data of the device:

- Direct access to the IEC address
- Access to IEC address via AT declaration
- Definition of an ALIAS for an IEC address (variable in IO mapping, e.g. IN0003\_I.ValueDigital)
- Link a program variable to an IEC address (IO mapping)

Information on safety applications: → **Access to safety input data and safety output data** (→ p. [136](#))

### 8.3.5 Default behaviour of the inputs and outputs in case of an error

25245



#### The following applies to inputs:

By default, automatic deactivation for **excessed measuring range** is enabled → Technical data. The corresponding channel will be set to the safe substitute value (0 / FALSE) in case of an error.

- For safety programming → **Safety programming** (→ p. [138](#))
- The automatic deactivation can be configured using the FB **ConfigDiagProt** (→ p. [294](#)).
- The threshold values for wire break and short-circuit detection can be configured using the FB **ConfigDiagLevel** (→ p. [290](#)).

#### For outputs, the following applies:

By default, automatic deactivation in case of **stuck-at-high** and **excessive current** is enabled → Technical data. The corresponding channel switches off automatically in case of a fault.

- For safety programming → **Safety programming** (→ p. [138](#))
- The automatic deactivation can be configured using the FB **ConfigDiagProt** (→ p. [294](#)).
- The threshold values for wire break and short-circuit detection can be configured using the FB **ConfigDiagLevel** (→ p. [290](#)).

### 8.3.6 Standard diagnostic limit values of inputs

60512

In the following table, standard limit values of the diagnostic/error detection are indicated in the operating modes:

Operating mode	DetectionTime	Lower limit value	Upper limit value
IN_DIGITAL_CSI	10 ms	0 % of VBB30	400 % of VBB30
IN_DIGITAL_CSI_NAMUR	10 ms	1 V	95 % of VBB30
IN_DIGITAL_CSO	10 ms	0 % of VBB30	100 % of VBB30
IN_VOLTAGE_10, IN_VOLTAGE_32, IN_CURRENT_CSI	10 ms	0	maximum measuring range of the set operating mode
IN_VOLTAGE_RATIO	10 ms	0	1000 ‰

Changing the diagnostic limits and the DetectionTime with the FB **ConfigDiagLevel** (→ p. [290](#)) or in the safety application with the FB **SF\_Input** (→ p. [322](#)).

The following operating modes have no preset diagnostic functionality:

- IN\_DIGITAL\_CSI\_BLANKING
- IN\_COUNT\_CSI
- IN\_COUNT\_CSO
- IN\_FREQUENCY\_CSI
- IN\_FREQUENCY\_CSO
- IN\_INC\_ENCODER\_CSI
- IN\_INC\_ENCODER\_CSO
- IN\_PERIOD\_RATIO\_CSI
- IN\_PERIOD\_RATIO\_CSO
- IN\_PERIOD\_RATIO\_US\_CSI

- IN\_PERIOD\_RATIO\_US\_CSO
- IN\_PHASE\_CSI
- IN\_PHASE\_CSO



The diagnostics for these operating modes can be implemented in the application, e.g. by evaluating the parameter `uiValueTime` (time elapsed since the last signal change at the input) on the corresponding function blocks.

### 8.3.7 Standard diagnostic limit values of outputs

60513

In the following table, standard limit values of the diagnostic/error detection are indicated in the operating modes:

Operating mode	DetectionTime	Lower limit value	Upper limit value
OUT_DIGITAL_CSO OUT_PWM_CSO OUT_CURRENT_CSO OUT_H_BRIDGE OUT_SENSOR_05 OUT_SENSOR_10	10 ms	0 mA	→ maximum switching current in data sheet

Changing the diagnostic limit values and the DetectionTime with the FB **ConfigDiagLevel** (→ p. 290) or in the safety application with the SF\_`[type]`Enh blocks.



In OUT\_PWM\_CSO, OUT\_CURRENT\_CSO and OUT\_H\_BRIDGE modes, the `uiDetectionTime` must be at least twice the PWM frequency. If dithering is used, 2 times the dither frequency must be used.

Example:

Frequency = 100 Hz

$uiDetectionTime(min) = 1 / 100 \text{ Hz} * 2 = 20 \text{ ms}$

The following operating modes have no preset diagnostic functionality:

- OUT\_DIGITAL\_CSI
- OUT\_PWM\_CSI
- OUT\_ANALOG\_10

### 8.3.8 Call sequence for diagnostic function blocks for inputs and outputs

25357

The call sequence for diagnostic function blocks for inputs and outputs must be programmed as follows:

#### Inputs:

1. FB **Input** (→ p. 270)
2. FB **ConfigDiagProt** (→ p. 294) (optional)
3. FB **ConfigSwThreshold** (→ p. 253) (optional)
4. FB **ConfigDiagLevel** (→ p. 290) (optional)

**Outputs:**

1. FB **Output** (→ p. [273](#)) / **PWM1000** (→ p. [312](#)) / **CurrentControl** (→ p. [309](#))
2. FB **ConfigDiagProt** (→ p. [294](#)) (optional)
3. FB **ConfigDiagLevel** (→ p. [290](#)) (optional)



- Do not execute the next block until the previous block has been completely processed (xPrepared = TRUE or xDone = TRUE).

## 8.4 Using ifm function libraries

39612

ifm electronic provides the following function libraries for the programming of the device under CODESYS 3.5:

Name	Description
ifmCANOpenManager	Functions for the use of the CAN interfaces as CANOpen Manager
ifmDeviceCR7xxS	Data structures, enumeration types and global variables
ifmFastInput	Functions to access the fast inputs of the device
ifmIOcommon	Functions for access to the inputs and outputs of the device
ifmIOconfigDiagProt	Functions to configure the I/O-related diagnostic and protective functions
ifmOutGroup	Functions to control output group switches
ifmOutHBridge	Functions to access H-bridge outputs
ifmOutPWM	Functions to access PWM outputs
ifmRawCAN	Functions for use of the CAN interfaces as CAN Layer 2
ifmSysInfo	Functions to set / read system information
ifmTypes	Global types and interfaces for other ifm libraries



Detailed information about the ifm function libraries: → **ifm function libraries** (→ p. [207](#))

### 8.4.1 Access to inputs

39529

To access the inputs of the device, the following functional elements are available:

Function element	Short description
<b>Input</b> (→ p. <a href="#">270</a> )	Assigns an operating mode to an input channel Provides the current state of the selected channel
<b>FastCount</b> (→ p. <a href="#">257</a> )	Counter block for fast input pulses
<b>IncEncoder</b> (→ p. <a href="#">260</a> )	Up/down counter function to evaluate encoders
<b>Period</b> (→ p. <a href="#">263</a> )	<ul style="list-style-type: none"> <li>measures at the indicated channel: the frequency and the period length (cycle time) in [μs],</li> <li>measures at the indicated channel pair: the phase shift in [°] between channel A and channel B</li> </ul>



## 8.4.2 Access to outputs

39530

To access the outputs of the device, the following functional elements are available:

Function element	Short description
<b>Output</b> (→ p. <a href="#">273</a> )	Assigns an operating mode to an output channel Provides the current state of the selected channel
<b>OutputGroup</b> (→ p. <a href="#">299</a> )	controls the activation status of an output group and provides diagnostic information about the group and the connected outputs. Using the FB, an output group including the corresponding outputs can be switched on or off.
<b>HBridge</b> (→ p. <a href="#">304</a> )	H bridge on a PWM channel pair
<b>PWM1000</b> (→ p. <a href="#">312</a> )	Initialises and configures a PWM-capable output channel the mark-to-space ratio can be indicated in steps of 1 %
<b>CurrentControl</b> (→ p. <a href="#">309</a> )	Current controller for a PWMi output channel

## 8.4.3 Control device

39480

The following function elements are available to control the device:

Function element	Short description
<b>SupplySwitch</b> (→ p. <a href="#">278</a> )	Switch off the unit
<b>SetLED</b> (→ p. <a href="#">276</a> )	Change the frequency and the colour of the status LED in the application program

## 8.4.4 Read device information

39669

To read information from the device the following functional elements are available:

Function element	Short description
<b>SystemSupply</b> (→ p. <a href="#">280</a> )	indicates the value of the system voltage
<b>Temperature</b> (→ p. <a href="#">282</a> )	indicates the value of the system temperature

## 8.5 Use IO mapping

### Content

Access inputs .....	118
Access outputs .....	119
Accessing the system inputs .....	119
Accessing the system outputs .....	120
Accessing user LEDs .....	121

25230

During the IO mapping (I/O image), global variables are coupled to the IEC addresses (e.g. %Ixx, %Qxx). The user can easily access the following elements from the application:

- inputs and outputs
- Functions of the display elements
- States of system components and characteristic values



The addresses of the system flags can change if the PLC configuration is extended.

- While programming only use the symbol names of the system flags!

### 8.5.1 Access inputs

25321

The user can use the following global variables to access the values of the inputs of the device.

Variable	Data type	Access	Description	Possible values	
INnnnn_I.					
ValueAnalogue	UINT	r	Value of the analogue input	0 ... 65535	
ValueDigital	BIT	r	Value of the digital input	FALSE	Input deactivated
				TRUE	Input activated
COM_RESET_I.					
ValueDigital	BIT	r	Communication reset → <b>Interface configuration file</b> <b>comconf.cfg</b> (→ p. <a href="#">104</a> )	FALSE	The settings of the file comconf.cfg are enabled.
				TRUE	The factory settings of the interfaces are enabled.

Legend:

r = read only

r/w = read and write



The valid value ranges and the type and number of the variables of the input depend on the active operating mode of the input.

- Please note the input configuration! → **Configure inputs and outputs** (→ p. [105](#))

## 8.5.2 Access outputs

39528

The user can use the following global variables to access the operating modes and the values of the outputs of the device.

Variable	Data type	Access	Description	Possible values	
OUTnnnn_I					
OutCurrent	UINT	r/w	Current value of the analogue output	0 ... 65535	
OutState	BIT	r/w	Output status	0 ... 4294967295	
OUTnnnn_Q					
ValueDigital	UINT	r/w	Digital output value	0 ... 65535	

Legend:

r = read only

r/w = read and write



The valid value ranges and the type and number of the variables of the output depend on the active operating mode of the output.

► Observe configuration of the outputs! → **Configure inputs and outputs** (→ p. [105](#))

## 8.5.3 Accessing the system inputs

25323

The user can use the following global variables to access the system inputs of the device:

Name	Data type	Access	Description	Possible values	
TemperatureBoard0					
Temperature	INT	r	Temperature on the system board (value in °C)	-32768 ... 32767	-32768 °C ... 32767 °C
Error	BIT	r	Error	FALSE	no error
				TRUE	Error
TemperatureChip0					
Temperature	INT	r	Chip temperature (value in °C)	-32768 ... 32767	-32768 °C ... 32767 °C
Error	BIT	r	Error	FALSE	no error
				TRUE	Error
VBB30					
VBBx	UINT	r	Voltage at power input VBB30 (value in mV)	0 ... 65535	0 mV ... 65535 mV
Error	BIT	r	Error	FALSE	no error
				TRUE	Error


VBB15					
VBBx	UINT	r	Voltage at power input VBB15 (value in mV)	0 ... 65535	0 mV ... 65535 mV
Error	BIT	r	Error	FALSE	no error
				TRUE	Error

Legend:  
r = read only

## 8.5.4 Accessing the system outputs

25328

The user can use the following global variables to access the system outputs of the device:

Name	Data type	Access	Description	Possible values	
VBBn_SW_Q					
ValueDigital	BIT	r/w	Activation request for the output group n (with n=0..5)	FALSE	Deactivate output group
				TRUE	Activate output group
VBBn_SW_I					
GroupCurrent	UINT	r	Measured output current of the entire group in [mA]	0 ... 65535	0 mA ... 65535 mA
VBBxVoltage	UINT	r	Measured voltage before the group switch in [mV]	0 ... 65535	0 mV ... 65535 mV
GroupVoltage	UINT	r	Measured voltage after the group switch in [mV]	0 ... 65535	0 mV ... 65535 mV
GroupState	UINT	r	Return value activation state of the selected output group  The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output group is deactivated
				TRUE	Output value is activated
Error	BIT	r	Error	FALSE	
				TRUE	

Legend:  
r = read only  
r/w = read and write

## 8.5.5 Accessing user LEDs

54373

The user can use the following global variables to access the values of the outputs of the device.

Variable	Data type	Access	Description	Possible values	
UserLEDn_I					
Error	BIT	r	Error bit of the LED	0	No error
				1	Error.
UserLED n_Q					
Value	BIT	r/w	Switch LED on/off	0	Switch LED off
				1	Switch LED on
Colour_1	Enum INT	r/w	LED colour status 1	→ <b>LED_COLOUR (ENUM)</b> (→ p. <a href="#">213</a> )	
Colour_2	Enum INT	r/w	LED colour status 0	→ <b>LED_COLOUR (ENUM)</b> (→ p. <a href="#">213</a> )	
Frequency	Enum INT	r/w	LED flashing frequency	→ <b>LED_FLASH_FREQ (ENUM)</b> (→ p. <a href="#">213</a> )	

Legend:

r = read only

r/w = read and write



When using an LED via the IO mapping on the safety PLC:

Access to the IO mapping must be from a SafePRG. Otherwise, the result will be a memory access violation.

## 8.6 Use RawCAN (CAN Layer 2)

### Content

RawCAN: Control CAN network nodes .....	122
RawCAN: Send and receive CAN messages.....	122
RawCAN: Request and send remote CAN messages .....	122

39608



► Observe the notes on task configuration! (→ **Configure task processing** (→ p. [86](#)))

In order to access one of the CAN interfaces configured for CANopen operation, the following POUs are available.

#### Requirements:

- The CAN interface is configured for operation as RawCAN (CAN Layer 2) (→ **Configuring CAN interfaces** (→ p. [96](#))).

### 8.6.1 RawCAN: Control CAN network nodes

39663

The following POUs are available to control a node in a CAN network:

Function element	Short description
<b>CAN_Enable</b> (→ p. <a href="#">217</a> )	initialises the specified CAN interface configures the CAN baud rate
<b>CAN_Recover</b> (→ p. <a href="#">219</a> )	controls the processing of a failure of the specified CAN channel If the CAN channel fails, reset the CAN interface and reboot

### 8.6.2 RawCAN: Send and receive CAN messages

39665

The following POUs are available to send or receive messages in a CAN network:

Function element	Short description
<b>CAN_Rx</b> (→ p. <a href="#">225</a> )	configures a data receive object and reads the receive buffer of the data object
<b>CAN_RxMask</b> (→ p. <a href="#">228</a> )	receives CAN messages of a non-coherent area The area is defined via a bit pattern and a bit mask
<b>CAN_RxRange</b> (→ p. <a href="#">231</a> )	receives CAN messages of a coherent area The area is defined via an upper and lower limit
<b>CAN_Tx</b> (→ p. <a href="#">240</a> )	asynchronous transmission of CAN messages

### 8.6.3 RawCAN: Request and send remote CAN messages

39664

The following POUs are available to request remote messages in a CAN network or to send replies to a remote request:

Function element	Short description
<b>CAN_RemoteRequest</b> (→ p. <a href="#">221</a> )	Send a request for a remote message
<b>CAN_RemoteResponse</b> (→ p. <a href="#">223</a> )	reply to the request of a remote message

## 8.7 Use CANopen

### Content

CANopen: Send and receive SDO .....	123
CANopen: Network Management (NMT) .....	123

23544

25134



► Observe the notes about task configuration! (→ **Configure task processing** (→ p. [86](#)))

In order to access one of the CAN interfaces configured for CANopen operation, the following POU's are available in **ifm** libraries.

Further POU's are available in CODESYS libraries by CODESYS GmbH.

### Requirements

- Device configured as CANopen Manager (master)  
→ Chapter **Configuring CAN interfaces** > **via system configuration: CANopen Manager** (→ p. [97](#))

### 8.7.1 CANopen: Send and receive SDO

39560

The following POU's are available to send or receive Service Data Objects (SDO):

Function element	Short description
<b>COP_SDOread</b> (→ p. <a href="#">246</a> )	Read Service Data Object (SDO)
<b>COP_SDOwrite</b> (→ p. <a href="#">248</a> )	Write Service Data Object (SDO)

### 8.7.2 CANopen: Network Management (NMT)

39559

The following POU's are available for the management of the CANopen network:

Function element	Short description
<b>COP_GetNodeState</b>	Request state of one or several CANopen devices
<b>COP_SendNMT</b> (→ p. <a href="#">250</a> )	Send an NMT control command to a CANopen device

## 8.8 Use SAE J1939

23802  
25134

- Observe the notes about task configuration! (→ **Configure task processing** (→ p. [86](#)))

To use the network protocol SAE J1939, the configuration is set via the device tree. (→ Chapter **Configuring CAN interfaces** > **via system configuration: J1939 manager** (→ p. [102](#)))  
CODESYS GmbH provides the IoDrvJ1939 with additional functions.



## 8.9 Using Ethernet

### Content

Modbus .....	126
--------------	-----

60514

60515

The SysSocket interface provides the following sockets for implementation in the IEC application:

Sockets	Number	Ports
TCP/IP	32	user-defined
UDP/IP	8	user-defined

The following CODESYS functions reserve the following number of sockets:

CODESYS function	Number of sockets
Transmitter network variable list	1
Receiver network variable list	1
TCP connection with the CODESYS protocol (e.g. Data Source Manager)	2
UDP connection with the CODESYS protocol (e.g. Data Source Manager)	1
In Modbus TCP master mode, per Modbus slave	1
In Modbus TCP slave mode	1

The following sockets and ports are reserved for the system:

Sockets	Number	Ports
TFTP (within the system)	1	69
CODESYS communication UDP (within the system)	1	1740, 1741, 1742, 1743
CODESYS communication TCP (within the system)	2	11740



► Do not use the reserved ports for TFTP and CODESYS in the application!

The implemented Ethernet stack is an embedded stack with limited resources. There is only a defined number of available send/receive buffers. These buffers are shared by all sockets in the system. This means that only a part of the total available sockets can be used for the IEC application to ensure system function, e.g. update and CODESYS debugging.

Available buffers:

Buffers	Maximum number
Send buffers	12
Receive buffers	32



- To ensure the system function, please observe the following:
- Recommendation: Only use a maximum of 16 receive sockets in the IEC application.
  - When the send buffer is full, the SysSocket function will return ERR\_SOCK\_NOBUFFER. Respond to this event in the IEC application, e.g. by resending the message.
  - The command FIONBIO of the function SysSocketIoctl only supports the call as non-blocking.
  - When the RESET COM pin is enabled, the SysSocketSetIPAddress and SetSubnetMask functions report the error ERR\_SOCK\_NOTSOCKET.
  - The SysSocket library may be used in the standard PLC and in the NonSafe-PRG of the Safety PLC.
  - The UDP Segmented Telegram is not supported. If more than 1460 bytes of data are sent, the application must split them into several UDP packets.
  - The CODESYS library SysSocket is released for use in the IEC application.
  - The CODESYS library UDP is intended exclusively for CODESYS internal use.
  - The CODESYS library TCP is intended exclusively for CODESYS internal use.



The following SysSocket functions have a timeout parameter and block the IEC task for the defined timeout period: SysSockRecvSizeUdp, SysSockPing, SysSockSelect.

A timeout period that is set too long will trigger the watchdog and lead to FATAL\_ERROR.

- Set the timeout period considering the task interval and the watchdog time: Timeout time < watchdog time < task interval

## 8.9.1 Modbus

60516

Information on Modbus connection:

- Observe the release notes.

## 9 Programming of the safety application

### Content

Objects in a safety PLC application.....	127
Using libraries that are and that are not from ifm .....	128
Error in the IEC application.....	128
Creating a safety PLC application .....	129
Safety programming .....	138
1-channel safety concept for inputs .....	140
2-channel safety concept for inputs.....	150
Safety concept of the outputs .....	163
Use CANopen-Safety .....	176

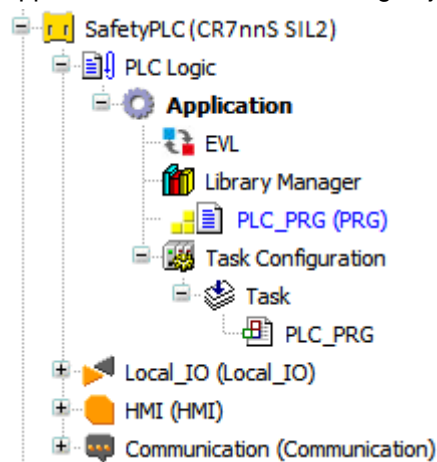
24525

The procedures described in this chapter refer to safety applications with a maximum safety level as described in chapter **Safety architecture** (→ p. 22).

### 9.1 Objects in a safety PLC application

24540

All objects of a safety PLC application are listed as sub-elements of the node [SafetyPLC (CR7xxS SIL2)] > [PLC Logic] > [Application] in the device tree. In the basic configuration, a standard PLC application includes the following objects:



[Application]	Container for objects of a PLC applications
[EVL]	Exchange Variable List. Contains variables the standard PLC is provided with via a shared memory area. → <b>Data exchange between standard PLC and safety PLC</b> (→ p. 92)
[Library Manager]	Provides access to standard, safety and device-specific function libraries: → <b>Using ifm function libraries</b> (→ p. 116) → <b>Safety programming</b> (→ p. 138)
[PLC_PRG(PRG)]	Provides access to the editor of the safety PLC application → <b>Creating a safety PLC application</b> (→ p. 129) The yellow symbol signifies a safety-related program.
[Task Configuration]	Provides access to the settings of the task processing: → <b>Configuration of the safety task processing</b> (→ p. 89)

If necessary, the user can add further objects to the safety PLC application.

## 9.2 Using libraries that are and that are not from ifm

25135

In a safety application with CODESYS Safety SIL2, only libraries that have the feature SIL2 (data type: BOOL) may be used.

- If SIL2 = TRUE: The library can be used without restriction in a safety-related application.
- If SIL2 = FALSE: The library can be integrated, but it may only be used in non-safe programs (NonSafe PRGs).
- If a library does not feature SIL2: The library may not be used in a safety application.

Libraries with the feature SIL2 may use libraries that do not have this feature and that have been sufficiently validated.

To use a library as already validated library within a safety application, the verification on the target hardware must be done with the same compiler settings with which the application will later run. To ensure that the compiler settings are identical, the same device description that are also used in the productive machine must be used for the device.

If the a library without the feature SIL2 is used in the safety application, this will lead to a warning during the compilation. For debugging purposes, this warning can be ignored.

## 9.3 Error in the IEC application

25159

The operating system (ifmOS) treats the following errors from the IEC application and leads the causing PLC into error class B. ( → **Error classes** (→ p. [462](#))



To improve the availability of the application, the project engineer should avoid the above-mentioned errors and avoid them e.g. by means of coding guidelines, review, test, etc.

Error in the application	Error class
Division by zero for Integer data type	B
Division by zero for Float data type	B
Memory access violation (writing)	B
Zero pointer exception	B

The operating system (ifmOS) has no error handling for the following arithmetic errors.



The project engineer must exclude the following errors by avoiding systematic errors and e.g. coding guidelines, review, test, etc. or allow the error calculations explicitly (e.g. subtraction of 2 values to get one absolute value).

Error in the application	Error class
Overflow (data types Integer and Float)	No error handling!
Underflow (data types Integer and Float)	No error handling!
Float Inexact Value	No error handling!

## 9.4 Creating a safety PLC application

### Content

Safety requirements .....	130
Supported programming languages .....	130
Complexity level / user level .....	131
Supported variable types .....	132
Access to safety input data and safety output data .....	136
Default behaviour of the inputs and outputs in case of an error .....	137

24527



Familiarise yourself with the following CODESYS functions!

- → Online help > CODESYS Development System > Programming of Applications
- → Online help > Add-ons > CODESYS Safety SIL2

When creating the project, CODESYS automatically generates the program block [PLC\_PRG (PRG)] under [SafetyPLC (CR7xxS SIL2)] > [PLC Logic] > [Application].

CODESYS highlights this block as [Safe PRG] (highlighted yellow in the device tree):  
→ [Properties] > [SIL2 Properties]

The safety PLC application may also include non safety-related standard program blocks (highlighted grey in the device tree). These blocks are used to execute non safety-related sub-functions in a safety-related application. The non safety-related sub-functions may not influence the safety-related functions (ensure absence of reaction!).

Only PRGs can be configured as non safety-related in the safety application. To implement the non safety-related functions or function blocks, use libraries that have the feature "SIL2" = FALSE. →

**Using libraries that are and that are not from ifm** (→ p. [128](#))

To create a safety PLC application:

- ▶ In the device tree:  
Double-click on [SafetyPLC (CR7xxS SIL2)] > [PLC Logic] > [Application] > [PLC\_PRG (PRG)].
- > The editor window shows the input mask of the selected programming language.
- ▶ Enter program code.

## 9.4.1 Safety requirements

24710



### WARNING

If the safety requirements are not complied with.

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function.
- ▶ When programming the safety PLC with CODESYS SIL2, adhere to the following documents:
  - → Safety requirements in chapter **Configuring the safety PLC** (→ p. 88) and in chapter **Programming of the safety application** (→ p. 127) of this manual. Observe the following mapping table:  
→ **Mapping table [H2] user manual / ifm ecomatController CR7xxS** (→ p. 483)
  - → [H2] User Manual CODESYS Safety SIL 2 V6.0 (CODESYS GmbH),  
→ [www.codesys.com](http://www.codesys.com)
  - → PLCopen TC5 Safety Software Technical Specification Part 1 to Part 4 (PLCopen), → [www.plcopen.org](http://www.plcopen.org)

## 9.4.2 Supported programming languages

58727



### WARNING

Use of an IEC programming language not certified by CODESYS for safety applications

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function.
- ▶ Only use the following IEC 61131 programming languages for safety programming:
  - Function block diagram FUP/FBD
  - Ladder diagram KOP/LD
  - Structured text ST with limited scope
- ▶ Note: → [H2] User Manual CODESYS Safety SIL 2 V6.0 (CODESYS GmbH),  
→ [www.codesys.com](http://www.codesys.com)

The following programming languages according to IEC 61131 may be used in non safety-related standard program blocks (gray marking in device tree):

- Function block diagram FUP/FBD
- Ladder diagram KOP/LD
- Structured text ST
- Sequential function chart AS/SFC
- Continuous function chart CFC

### 9.4.3 Complexity level / user level

24749

For the creation of safety-related applications for the device, various complexity levels / user levels are available:

1. Basic Level
2. Extended Level
3. System Level

By programming at a low level, the certification requirements of the software parts can be simplified. The complexity level results from the following parameters:

- from the applied programming language
- from the complexity that is programmed in it

Programming language		Complexity level		
		Basic Level	Extended Level	System Level
Function block diagram	FUP / FBD	X	X	X
Ladder diagram	KOP / LD	X	X	X
Sequential function chart	AS / SFC	–	–	X
Instruction list	AWL / IL	–	–	X
Continuous Function Chart	CFC	–	–	X
Structured text	ST	–	–	X

Legend:

X = is supported

– = is not supported

More information:

→ PLCopen TC5, Safety Software Technical Specification Part 1, Version 1.0  
 > Chapter 4.1 "Definition of User Levels"

→ [www.plcopen.org](http://www.plcopen.org)



Which validation and verification measures are required precisely for the individual levels depends on the security guidelines to be met.

## 9.4.4 Supported variable types

### Content

Safety data types .....	133
Non-volatile data .....	133
Function description of non-volatile data .....	134
Non-volatile safety-related data (static) .....	135
Non-volatile safety-related data (dynamic) .....	136

24746



► Familiarise yourself with the following CODESYS functions!

- Local variables  
→ Online help > CODESYS Development System > Reference, Programming > Variable Types and special Variables > Local Variables – VAR
- Global variable list  
→ Online help > CODESYS Development System > Reference, Programming > Variable Types and special Variables > Global Variables - VAR\_GLOBAL
- network variables UDP  
→ Online help > CODESYS Development System > Working with Control Networks > Network Variables
- Non-volatile variables - RETAIN PERSISTENT  
→ Online help > CODESYS Development System > Reference, Programming > Variable Types and special Variables > Remanent Variables - RETAIN, PERSISTENT

The device supports the following variable types:

Variable type	Declaration	Scope of validity	Application	Memory behaviour
local	In the declaration part of the POU	Applies only to the POU in which it has been declared	Standard / Safe	volatile
locally PERSISTENT				non volatile for: - reset warm/cold - download
global	In the global variable list (GVL)	Applies to all POUs of the project	Standard / Safe	Volatile
global PERSISTENT				non volatile for: - reset warm/cold - download
Network	In network variable lists	Values are available to all projects in the whole network if the variable is contained in their network variables lists. Only UDP network variables are supported.	Standard	volatile



For performance reasons, do not overuse 64 bit variables!



## Safety data types

58728

To clearly differentiate safety signals in the program organisation from standard signals, the prefix SAFE is defined to mark the safety data types, e.g. SAFEBOOL instead of BOOL.

Signals with safety data types and the safe signal flow are highlighted yellow in the editor.

This simplifies and shortens the signal flow verification.

Possible errors in the safety application are thereby minimised.

The following safety data types are available:

Data type	Min. value	Max. value	Length
SAFEBOOL	FALSE	TRUE	1 bit
SAFESINT	-128	127	8 bits
SAFEUSINT	0	255	8 bits
SAFEINT	-32768	32767	16 bits
SAFEUINT	0	65535	16 bits
SAFEDINT	-2147483648	2147483647	32 bits
SAFEUDINT	0	4294967295	32 bits
SAFELINT	-922372036854775808	922372036854775807	64 bits
SAFEULINT	0	18446744073709551615	64 bits
SAFETIME	T#0	T#49d17h2m37s296m	



- For the safety application, also the data types REAL and LREAL can be used in the system level.
- For the safety application, also the standard data types can be used in the system level.
  - **Complexity level / user level** (→ p. [131](#))
  - **Data types** (→ p. [109](#))

## Non-volatile data

60510

Different types of non-volatile data are supported:

- Retain variables / retain data (VAR\_PERSISTENT)
- Memory bytes (%MB, %MW, etc.)

Variables declared as VAR\_PERSISTENT generate non-volatile data.

Memory bytes are pre-defined non-volatile memory areas.

In the following, both types will be referred to as non-volatile data.

Non-volatile data includes the values that are stored in it during the following processes:

- Switching the device on/off
- Online reset (warm/cold) of the device
- Loading the PLC (download)

Typical applications for non-volatile variables are, for example:

- Operating hours that are counted up and retained while the machine is in operation,
- Position values of incremental encoders,
- Target values entered in the display unit,
- Machine parameters,

i.e. all variables whose values must not get lost when the device is switched off.

## Function description of non-volatile data

60511

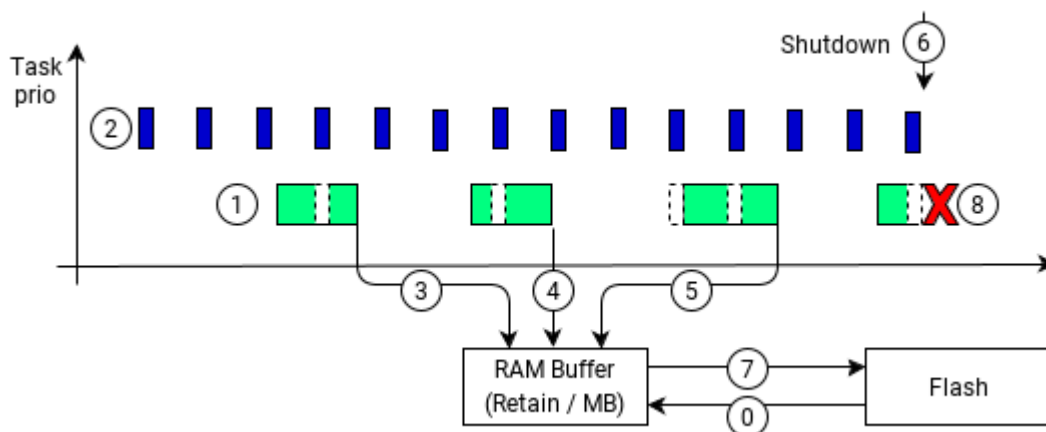
After power-on, the non-volatile data is written from the FLASH memory to the RAM intermediate buffer (0).

At the end of the cycle of the low-priority task (1), non-volatile data is written into an intermediate RAM buffer (3, 4, 5).

The execution of the lower priority task (1) is not interrupted by higher priority tasks (2) at the end of the cycle. This ensures the consistency of the non-volatile data.

In the SHUTDOWN state (6), the non-volatile data is written from the volatile RAM memory to the non-volatile flash memory (7).

Only the data of the last completely executed task will be stored in the RAM intermediate buffer (5). The data of an unfinished task (8) will not be stored.



### WARNING

Data loss in case of FATAL\_ERROR

- > Any non-volatile data in the RAM intermediate buffer that has changed since the application was started will be lost.
- > Personal injury and/or damage to property is possible.
- Measures to be taken depending on the application according to → **Non-volatile safety-related data (static)** (→ p. [135](#)) or → **Non-volatile safety-related data (dynamic)** (→ p. [136](#)) must be observed.



After occurrence of a FATAL\_ERROR followed by a Power-On reset of the controller, the data stored in the FLASH memory during the last SHUTDOWN will be available.



Please note the following information:

- → **Switch on/off via main switch** (→ p. [43](#))
- → **Switch on/off via ignition lock (clamp 15)** (→ p. [43](#))
- → **FB SupplySwitch** (→ p. [278](#))
- → **Operating states** (→ p. [198](#))
- → **Error classes** (→ p. [462](#))

## Non-volatile safety-related data (static)

54387

The controller is suited to store safety-related static data (that cannot be changed while the controller is operated).



### WARNING

Risk that static data will not be written into the non-volatile memory.

- > Personal injury and/or damage to property is possible.
- ▶ Check once if newly collected or modified data is correct after a subsequent power-on reset before releasing the overall application.

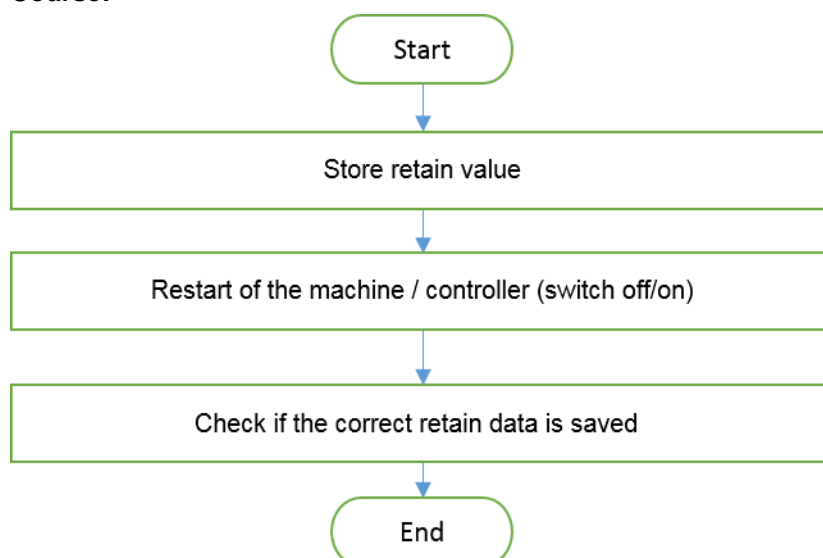
The following explanations refer to the storage of safety-related data in the non-volatile memory. This includes the following memory areas:

- non volatile variables (retain variables)
- Memory Bytes %MB

Using safety-related data in the non-volatile memory is only supported in a way that the data has been verified before application is delivered and will no longer be changed after the application is delivered. This means that safety-related data in the non-volatile memory will not be not changed while the machine is operated (operation at the end customers).

Whether the data has been properly stored in the non-volatile memory can, for example, be verified within the scope of the production or during the set-up by doing a functional test.

**Course:**



## Non-volatile safety-related data (dynamic)

60518

The controller is **not** suited to store safety-related dynamic data (that can be changed while the controller is operating) without additional measures in the application.



### WARNING

Data loss in case of FATAL\_ERROR.

- > Personal injury and/or damage to property is possible.
- ▶ Measures to be taken depend on the application, e.g. retain data must be verified again after each power-on reset of the controller by means of double storage using separate hardware and by comparing the data.
- ▶ Double data storage in the same controller is not sufficient.

## 9.4.5 Access to safety input data and safety output data

58729



Implement the access of the safety PLC to the input data and output data as described in → **Safety programming** (→ p. [138](#)).

The safety PLC may not access the input and output data via the IO mapping.

If no safety function blocks (SF) are used, the diagnostic messages of the used block must be treated by the application.

If unexpected messages occur in the block, these must lead to the safe state through the application.



Signals at safe inputs must change with sufficient frequency during operation:

- For digital inputs: Detection of the rising AND falling edge
- For analogue inputs: Detection of the value change > safety tolerance value of mission and monitor signal

In applications where this is not ensured, latent errors may occur, which can lead to a dangerous failure at the next demand upon the safety function(→ **Self-test at restart (start-up test)** (→ p. [28](#))). To avoid this situation, it is recommended to test inputs with rarely changing signals at suitable intervals, e.g. by

- performing a regular emergency off test
- dynamising the signal at the start of the application
- using a safety switch with a regular blanking pulse, → **Operation as fail-safe digital input with blanking pulses** (→ p. [146](#))

A suitable time interval for the execution of signal change must be derived from the safety concept for the application or the applicable product standards of the application.

A power-on reset (restart of the controller) does **not** lead to a valid signal change at the fail-safe inputs!

## 9.4.6 Default behaviour of the inputs and outputs in case of an error

25245



### The following applies to inputs:

By default, automatic deactivation for **excessed measuring range** is enabled → Technical data. The corresponding channel will be set to the safe substitute value (0 / FALSE) in case of an error.

- For safety programming → **Safety programming** (→ p. [138](#))
- The automatic deactivation can be configured using the FB **ConfigDiagProt** (→ p. [294](#)).
- The threshold values for wire break and short-circuit detection can be configured using the FB **ConfigDiagLevel** (→ p. [290](#)).

### For outputs, the following applies:

By default, automatic deactivation in case of **stuck-at-high** and **excessive current** is enabled → Technical data. The corresponding channel switches off automatically in case of a fault.

- For safety programming → **Safety programming** (→ p. [138](#))
- The automatic deactivation can be configured using the FB **ConfigDiagProt** (→ p. [294](#)).
- The threshold values for wire break and short-circuit detection can be configured using the FB **ConfigDiagLevel** (→ p. [290](#)).

## 9.5 Safety programming

### Content

Safety-related applications .....	138
-----------------------------------	-----

24548

For safety programming of the device, **ifm electronic** provides the following certified function libraries under CODESYS 3.5. The function blocks are specially designed for programming at basic and extended level.



Detailed information about the levels of complexity: → **Complexity level / user level** (→ p. [131](#))

Name	Description
ifmPLCopenSafe	Safety standard functions according to PLCopen specification
ifmIOSafety	Additional safety IO functions
ifmPLCopenAddonSafe	Additional safety functions to use input signals with standard data types in safety applications



Detailed information about the **ifm** function libraries: → **ifm function libraries** (→ p. [207](#))

### 9.5.1 Safety-related applications

24712

The following is available to implement safety-related applications:

- all function blocks of the CODESYS library Standard Library by CODESYS GmbH
- CODESYS standard operators and standard conversions, see → **Using the operators** (→ p. [478](#))
- the certified function blocks of the libraries ifmIOSafety and ifmPLCopenAddonSafe
- the following certified function blocks of the library ifmPLCopenSafe :

Function element	Short description
<b>SF_Antivalent</b> (→ p. <a href="#">394</a> )	Compares 2 digital fail-safe inputs while considering a configurable discrepancy time with regard to antivalence with each other.
<b>SF_CamshaftMonitor</b> (→ p. <a href="#">396</a> )	The function block monitors the camshaft. It monitors a defined number of signal changes for a set duration.
<b>SF_DoubleValveMonitoring</b> (→ p. <a href="#">400</a> )	Monitors the switching behaviour of fluidic double valves (safety valves for presses).
<b>SF_EDM</b> (→ p. <a href="#">404</a> )	Controls a safety output and monitors the controlled actuators.
<b>SF_EmergencyStop</b> (→ p. <a href="#">408</a> )	Monitors an e-stop.
<b>SF_EnableSwitch</b> (→ p. <a href="#">412</a> )	Evaluation of the fail-safe signals of a release switch with three switching stages.
<b>SF_Equivalent</b> (→ p. <a href="#">415</a> )	Compares 2 digital fail-safe inputs while considering a configurable discrepancy time with regard to equivalence with each other.
<b>SF_ESPE</b> (→ p. <a href="#">417</a> )	Monitors electro-sensitive protective equipment (ESPE), e.g. a light grid or a laser scanner.
<b>SF_FootSwitch</b> (→ p. <a href="#">421</a> )	Evaluates the signals of a foot switch with three switching positions according to DIN EN 60204 section 9.2.5.8.

Function element	Short description
<b>SF_GuardLocking</b> (→ p. <a href="#">424</a> )	Controls and monitors the access to a hazardous area by means of an interlocking guard with guard locking.
<b>SF_GuardMonitoring</b> (→ p. <a href="#">428</a> )	Monitors protective equipment with two-stage locking and configurable maximum monitoring time.
<b>SF_ModeSelector</b> (→ p. <a href="#">432</a> )	Enables reliable switching between up to 8 operating modes of a machine or installation.
<b>SF_OutControl</b> (→ p. <a href="#">435</a> )	Monitors a fail-safe output with a signal from the functional application and a signal from the fail-safe application.
<b>SF_SafetyRequest</b> (→ p. <a href="#">439</a> )	Provides an interface to a safety actuator, e.g. safety drive or safety valve with back channel.
<b>SF_SingleValveCycleMonitoring</b> (→ p. <a href="#">442</a> )	Cyclic monitoring of safety cartridge valves.
<b>SF_SingleValveMonitoring</b> (→ p. <a href="#">446</a> )	Monitors the switching behaviour of fluidic valves.
<b>SF_TwoHandControlTypeII</b> (→ p. <a href="#">450</a> )	Two-hand control type II
<b>SF_TwoHandControlTypeIII</b> (→ p. <a href="#">453</a> )	Two-hand control type III
<b>SF_ValveGroupControl</b> (→ p. <a href="#">456</a> )	Combination of the connected safety valves in a group with group switch-off in case of a fault.

## 9.6 1-channel safety concept for inputs

### Content

Operation as fail-safe digital input .....	140
Operation as fail-safe analogue input.....	143
Operation as fail-safe digital input with blanking pulses.....	146

24549

24743

Depending on the requirement, a safety-related input can have a 1-channel design and be reliably processed in the safety application. This is only admissible for the following input types and operating modes:

Input type	permissible operating mode (eMode) for safety operation	function block required for safety operation
IN MULTIFUNCTION-A	IN_DIGITAL_CSI IN_DIGITAL_CSI_NAMUR IN_VOLTAGE_10 IN_VOLTAGE_32 IN_VOLTAGE_RATIO	<b>SF_Input</b> (→ p. 322)
IN FREQUENCY-B	IN_DIGITAL_CSI_BLANKING	<b>SF_InputBlanking</b> (→ p. 327)

For 1-channel access to the fail-safe inputs of the device, the following function blocks are available:

Function element	Short description
<b>SF_Input</b> (→ p. 322)	<ul style="list-style-type: none"> <li>• assigns an operating mode to a 1-channel input channel</li> <li>• provides the current status of the selected channel as safe signal for further processing</li> </ul>
<b>SF_InputBlanking</b> (→ p. 327)	<ul style="list-style-type: none"> <li>• 1-channel reading of an input</li> <li>• verifies the input status by means of blanking signals and provides a safe signal for further processing</li> </ul>

### 9.6.1 Operation as fail-safe digital input

60520

The safety concept for IN\_MULTIFUNCTION\_A for operation as **fail-safe digital input** requires the following:

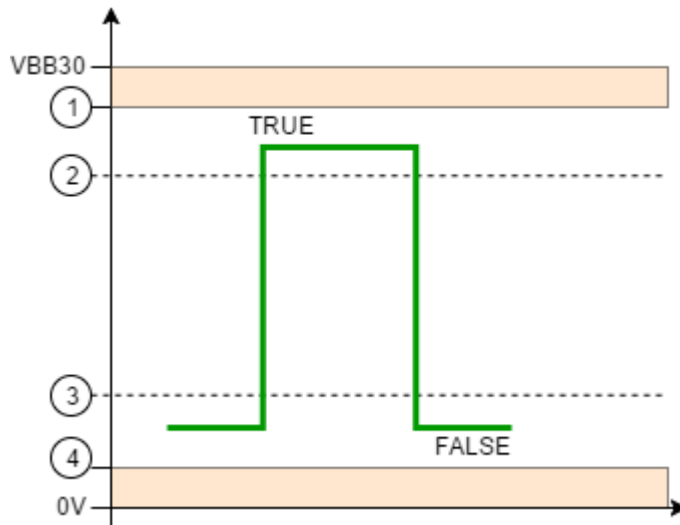
- Using the function block **SF\_Input** (→ p. 322) in the operating modes:
  - IN\_DIGITAL\_CSI
  - IN\_DIGITAL\_CSI\_NAMUR
- Definition of the valid signal range with the inputs uiDiagLimitMin (signal range minimum value) and uiDiagLimitMax (signal range maximum value) on the function block **SF\_Input** (→ p. 322)
- Definition of the time (detection time) from when the deviation from the valid signal range is detected as an error at the input uiDetectionTime of the function block **SF\_Input** (→ p. 322)
- Sufficiently frequent signal change.  
In applications where this is not ensured, latent errors may occur, which can lead to a dangerous failure at the next demand upon the safety function. To minimise the risk of a latent error, it is recommended to test inputs with rarely changing signals at suitable intervals, e.g. by
  - performing a regular emergency off test
  - dynamising the signal at the start of the application
  - Using a safety switch with regular blanking pulse, → **Operation as fail-safe digital input with blanking pulses** (→ p. 146)

A suitable time interval for the execution of signal change must be derived from the safety concept for the application or the applicable product standards of the application.



**Switching thresholds (standard setting) with hysteresis for both operating modes (→ datasheet):**

- Upper switching threshold (switching threshold\_H, FALSE → TRUE) at 70% of VBB30
- Lower switching threshold (switching threshold\_L, TRUE → FALSE) at 30% of VBB30



**Legend**

- |   |                            |
|---|----------------------------|
| 1 | Signal range maximum value |
| 2 | Switching threshold_H      |
| 3 | Switching threshold_L      |
| 4 | Signal range minimum value |

The following **errors within the safety controller** are detected when using the FB **SF\_Input** (→ p. 322) if the error lasts at least as long as the set `uiDetectionTime` :

- Wire break / interruption of the signal
- Short circuit of the signal with another signal (cross fault)
- Short circuit to supply voltages VBBnn
- Short circuit with GND
- Drift (in the signal path)
- Stuck-at-low (signal is stuck at FALSE)
- Stuck-at-high (signal is stuck at TRUE)

If required by the safety concept of the application, the following **errors outside the controller** must be prevented or recognised by means of additional measures of the application:

- Short circuit to other (sensor) cables
- Short circuit to supply voltages VBBnn
- Short circuit to GND

To detect the **external errors**, the input signal must be restricted by means of suitable user-specific limitation of the signal range (setting the minimum and maximum values of the signal range). Then, also the following **errors outside the safety controller** can be detected within the time set at the input `uiDetectionTime` of the function block **SF\_Input** (→ p. 322) :

- Short circuit to supply voltages VBBnn
- Short circuit to GND
- Wire break / interruption of the signal

25337



## WARNING

A dangerous short circuit to VBBnn outside the controller will not be detected.

- > Risk of personal injuries and/or damage to property.
- > Requesting the safety function is not possible.
- ▶ Set the maximum value in accordance with the valid signal range at the input uiDiagLimitMax of the FB **SF\_Input** (→ p. [322](#))! Select uiDiagLimitMax < 1000 ‰ of VBB30 in any case!



If the following errors do not need to be detected in the application because the error corresponds with the safe state, the minimum value of the signal range (uiDiagLimitMin) can be set as 0 ‰:

- Short circuit to GND
- Wire break / interruption of the signal

The function block **SF\_Input** (→ p. [322](#)) has the following behaviour if an error is detected:

- The fail-safe outputs of the function block are put into the safe state (0 and FALSE)
- The error code can be read at the output DiagCode



→ Detailed information about the FB **SF\_Input** (→ p. [322](#))



Only the outputs, groups and messages that are chained in the safety function will be switched off by the IEC application. Other outputs, groups and messages are not affected by this.



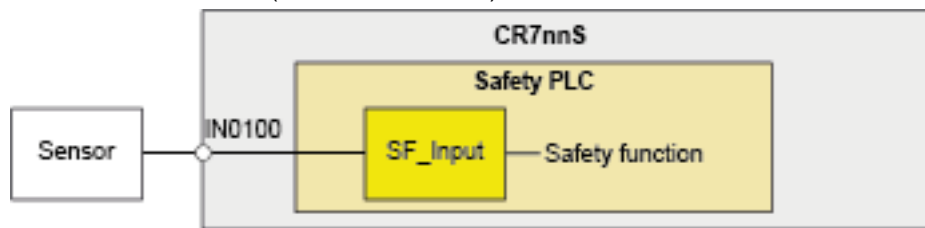
For SISTEMA:

- ▶ Use the subsystem input one-channel in the ifm VDMA library.

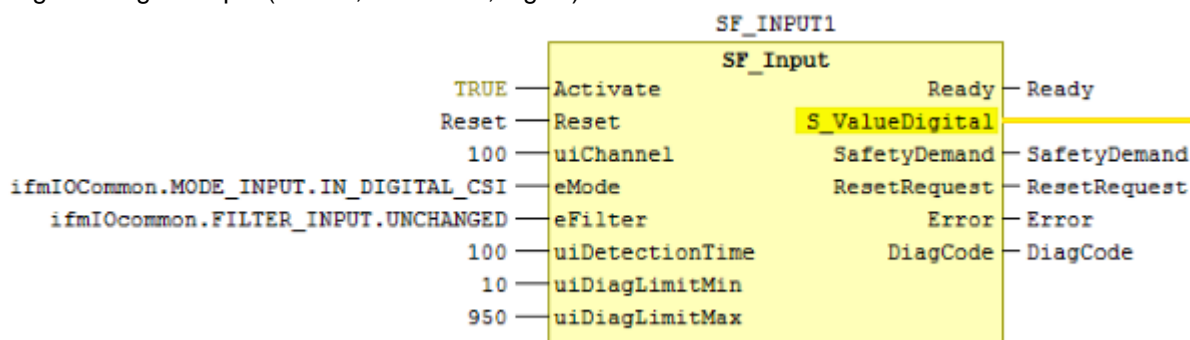
## Example

24541

Schematic illustration (sensor, 1-channel):



Programming example (sensor, 1-channel, digital):



### 9.6.2 Operation as fail-safe analogue input

60521

The safety concept for IN\_MULTIFUNCTION\_A for operation as **fail-safe analogue input** requires the following:

- Using the function block **SF\_Input** (→ p. 322) in the operating modes:
  - IN\_VOLTAGE\_10
  - IN\_VOLTAGE\_32
  - IN\_VOLTAGE\_RATIO
- Definition of the valid signal range with the inputs uiDiagLimitMin (signal range minimum value) and uiDiagLimitMax (signal range maximum value) on the function block **SF\_Input** (→ p. 322)
- Definition of the time (detection time) from when the deviation from the valid signal range is detected as an error at the input uiDetectionTime of the function block **SF\_Input** (→ p. 322)
- Sufficiently frequent signal change.  
In applications where this is not ensured, latent errors may occur, which can lead to a dangerous failure at the next demand upon the safety function. To minimise the risk of a latent error, it is recommended to test inputs with rarely changing signals at suitable intervals, e.g. by
  - dynamising the signal at the start of the application
  - Do not supply the analogue sensor with power until the controller has been started

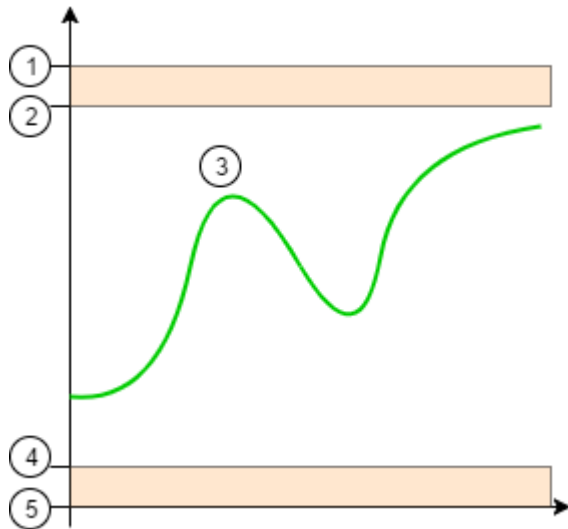
A suitable time interval for the execution of signal change must be derived from the safety concept for the application or the applicable product standards of the application.

#### Signal range for the individual operating modes:

Measuring range for the individual operating modes: → Data sheet

To detect all errors, the analogue signal must be within the valid signal range between the minimum value and the maximum value:

Mode	Minimum	Max.	Safety tolerance
IN_VOLTAGE_10	> 310 mV	10 V	± 310 mV
IN_VOLTAGE_32	> 740 mV	32 V	± 740 mV
IN_VOLTAGE_RATIO	> 740 mV	32 V	± 740 mV



#### Legend

- 1 Measuring value maximum range
- 2 Signal range maximum value
- 3 Analogue signal
- 4 Signal range minimum value
- 5 0 [V / mA / %]

The following **errors (signal deviation greater than the safety tolerance) within the safety controller** are detected when using the FB **SF\_Input** (→ p. [322](#)) if the error has been present for at least as long as the set `uiDetectionTime` :

- Wire break / interruption of the signal
- Short circuit of the signal with another signal (cross fault)
- Short circuit to supply voltages VBBnn
- Short circuit with GND
- Drift (in the signal path)
- Stuck-at (signal is stuck at one value)

**If required by the safety concept of the application, the following errors outside the controller must be prevented or recognised by means of additional measures of the application:**

- Short circuit to other (sensor) cables
- Short circuit to supply voltages VBBnn
- Short circuit to GND

To detect the external errors, the input signal must be restricted by means of appropriate application-specific limitation of the value range (setting the minimum and the maximum value). Then, also the following errors can be detected outside the safety controllers within the **uiDetectonTime** set at the function block **SF\_Input** (→ p. 322):

- Short circuit to supply voltages VBBnn
- Short circuit to GND
- Wire break / interruption of the signal

25340



## WARNING

A dangerous short circuit to VBBnn outside the controller will not be detected.

- > Risk of personal injuries and/or damage to property.
- > Requesting the safety function is not possible.
- ▶ Set the maximum value in accordance with the valid signal range at the input **uiDiagLimitMax** of the FB **SF\_Input** (→ p. 322)! In any case, select **uiDiagLimitMax** < upper limit of the measuring range (→ Data sheet)!



If the following errors do not need to be detected in the application **and** within the controller because the error corresponds with the safe state, the minimum value of the signal range (**uiDiagLimitMin**) can be set as 0 %:

- Short circuit to GND
- Wire break / interruption of the signal



If the detection of the errors occurs too fast due to short interference, this can be configured using the input **uiDetectionTime**. This increases the error detection time inside and outside the controller.

The function block **SF\_Input** (→ p. 322) has the following behaviour if an error is detected:

- The fail-safe outputs of the function block are put into the safe state (0)
- The error code can be read at the output **DiagCode**



→ Detailed information about the FB **SF\_Input** (→ p. 322)



Only the outputs, groups and messages that are chained in the safety function will be switched off by the IEC application. Other outputs, groups and messages are not affected by this.



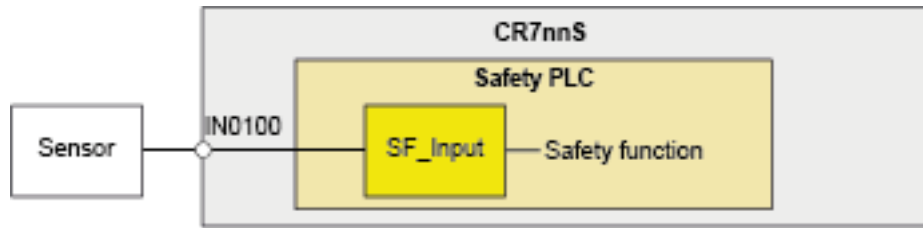
For SISTEMA:

- ▶ Use the subsystem input `one-channel` in the `ifm VDMA` library.

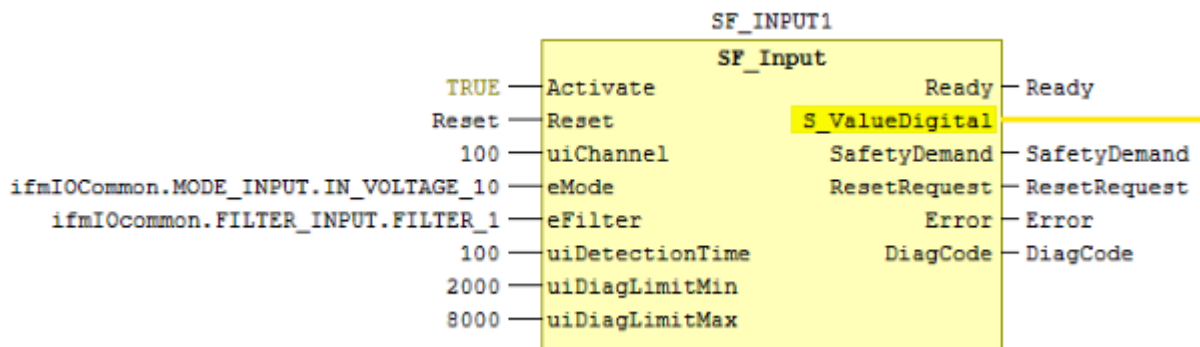
## Example

25365

Schematic illustration (sensor, 1-channel):



Programming example (sensor, 1-channel, analogue):



### 9.6.3 Operation as fail-safe digital input with blanking pulses

25188

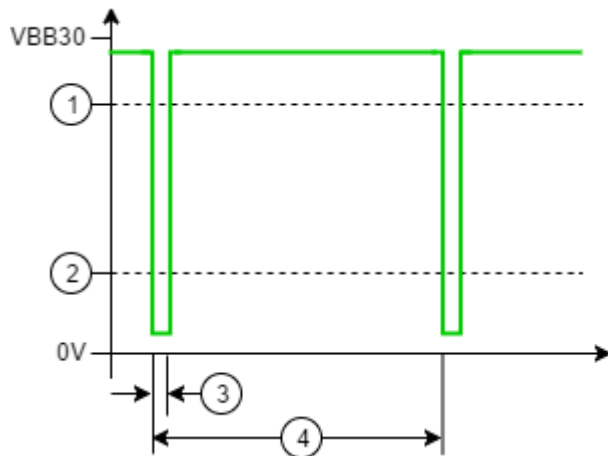
The safety concept for IN\_FREQUENCY\_B for operation as **fail-safe digital input with blanking pulses** requires the following:

- FALSE is the safe state of the input
- Using the function block **SF\_InputBlanking** (→ p. 327)
- Determining the distance (4) of the blanking pulses with the input udiBlankingTimeMax of the function block **SF\_InputBlanking** (→ p. 327)
- Set udiBlankingTimeMin to 0
- Minimum pulse width (3)  $\geq 10 \mu\text{s}$
- Maximum pulse width (3)  $\leq 800 \mu\text{s}$
- The pulse must be below the switching threshold\_L ( $< 30\%$  of VBB30)
- Minimum pulse spacing (4)  $\geq 16 \text{ ms}$

**Switching thresholds (standard setting) with hysteresis for the operating modes (→ datasheet):**

- Upper switching threshold (switching threshold\_H, FALSE -> TRUE) at 70% of VBB30

- Lower switching threshold (switching threshold\_L, TRUE -> FALSE) at 30% of VBB30



#### Legend

- |   |                       |
|---|-----------------------|
| 1 | Switching threshold_H |
| 2 | Switching threshold_L |
| 3 | Pulse width           |
| 4 | Pulse spacing         |

The following **errors** are detected when using the function block **SF\_InputBlanking** (→ p. 327) within the time (Application Task Cycle + Blanking Time Max) (→ **Time-related behaviour IEC application** (→ p. 33)):

- Short circuit to supply voltages VBBnn
- Stuck-at-high (signal is stuck at TRUE)

**The following errors are NOT detected, but correspond with the fail-safe state (input = FALSE):**

- Wire break / interruption of the signal
- Short circuit with GND
- Stuck-at-low (signal is stuck at FALSE)

**If required by the safety concept of the application, the following errors outside the controller must be prevented or recognised by means of additional measures of the application:**

- Short circuit to other (sensor) cables
- Short circuit to supply voltages VBBnn



## WARNING

Wrong configuration of the blanking time (max).

- > Risk of personal injuries and/or damage to property.
- > Error detection not possible or too late.
- > Requesting the safety function is not possible.
- ▶ The user must set the blanking time (max) in accordance with the expected spacing of the input signal (4). This information depends on the sensor or the signal source that is used.
- ▶ The blanking time (max) has an impact on the safety time. Errors will not be detected before the blanking time (max) has elapsed.
- ▶ Calculating the safety time.
- ▶ Ensure that the safety time is sufficient to comply with the process safety time ( → **The process safety time** (→ p. 30))

The function block **SF\_InputBlanking** (→ p. 327) following behaviour if an error is detected:

- The fail-safe outputs of the function block **SF\_InputBlanking** (→ p. 327) are put into the safe state (0 and FALSE)
- The error code can be read at the output DiagCode



→ Detailed information about the FB **SF\_InputBlanking** (→ p. 327)



Only the outputs, groups and messages that are chained in the safety function will be switched off by the IEC application. Other outputs, groups and messages are not affected by this.

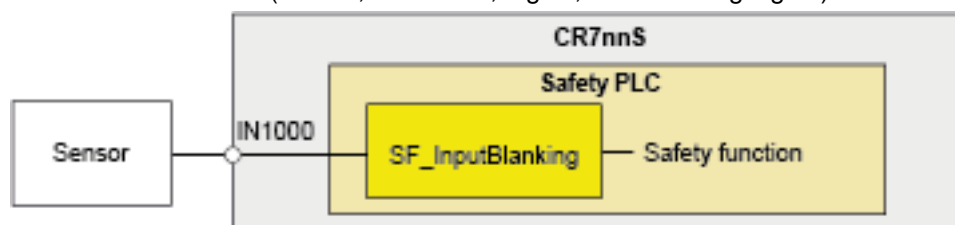


For SISTEMA:

- ▶ Use the subsystem input one-channel in the ifm VDMA library.

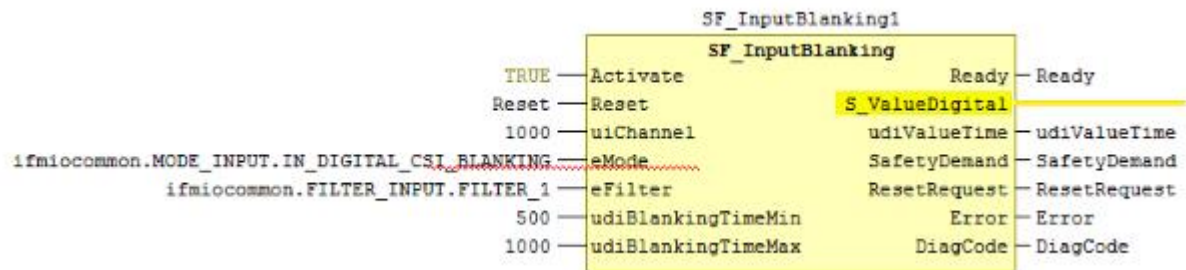
## Example

Schematic illustration (sensor, 1-channel, digital, with blanking signal):





Programming example:



## 9.7 2-channel safety concept for inputs

### Content

fail-safe 2-channel evaluation of the inputs .....	151
Operation as 2-channel fail-safe digital input .....	153
Operation as 2-channel fail-safe analogue input.....	156
Operation as 2-channel fail-safe frequency input.....	160

24850

25339

Depending on the requirement, a safety-related input can have a 2-channel design and be reliably processed in the safety application. This is only admissible for the following input types and operating modes:

Input type	permissible operating mode (eMode) for safety operation	function block required for safety operation
IN MULTIFUNCTION-A	IN_DIGITAL_CSI IN_DIGITAL_CSI_NAMUR IN_VOLTAGE_10 IN_VOLTAGE_32 IN_VOLTAGE_RATIO IN_CURRENT_CSI	<b>SF_Input</b> (→ p. <a href="#">322</a> )
IN FREQUENCY-B	IN_DIGITAL_CSI IN_DIGITAL_CSI_NAMUR IN_DIGITAL_CSI_BLANKING IN_VOLTAGE_10 IN_FREQUENCY_CSI IN_INC_ENCODER_CSI IN_PERIOD_RATIO_CSI IN_PERIOD_RATIO_US_CSI IN_PHASE_CSI IN_COUNT_CSI	<b>SF_InputBlanking</b> (→ p. <a href="#">327</a> )
IN RESISTOR-A	IN_DIGITAL_CSI IN_DIGITAL_CSI_NAMUR	<b>SF_Input</b> (→ p. <a href="#">322</a> )
IN DIGITAL-A/B	IN_DIGITAL_CSI IN_DIGITAL_CSI_NAMUR	<b>SF_Input</b> (→ p. <a href="#">322</a> )

If the specific application allows for it, standard input signals can be used in safety functions in the Basic Level and Extended Level.

► To do so, read the signals via 2 channels and compare them using a safety function block.

For this purpose, ifm offers the following safety function blocks:

Function element	Short description
<b>SF_Equivalent_BOOL</b> (→ p. <a href="#">375</a> )	<ul style="list-style-type: none"> <li>compares two digital inputs with each other whether they are identical</li> <li>verifies the time flow of the inputs with regard to each other</li> <li>provides the logic operation result as a safety signal</li> </ul>
<b>SF_Antivalent_BOOL</b> (→ p. <a href="#">377</a> )	<ul style="list-style-type: none"> <li>compares whether two digital inputs are non-identical</li> <li>verifies the time flow of the inputs with regard to each other</li> <li>provides the logic operation result as a safety signal</li> </ul>
<b>SF_Equivalent_DINT</b> (→ p. <a href="#">379</a> )	<ul style="list-style-type: none"> <li>compares two input values [DINT] with each other</li> <li>verifies the values with regard to the permissible value range and permissible deviation</li> <li>provides the result as a safety signal</li> </ul>
<b>SF_Equivalent_UINT</b> (→ p. <a href="#">382</a> )	<ul style="list-style-type: none"> <li>compares two input values [UINT] with each other</li> <li>verifies the values with regard to the permissible value range and permissible deviation</li> <li>provides the result as a safety signal</li> </ul>

Function element	Short description
<b>SF_Equivalent_UDINT</b> (→ p. 385)	<ul style="list-style-type: none"> <li>compares two input values [UDINT] with each other</li> <li>verifies the values with regard to the permissible value range and permissible deviation</li> <li>provides the result as a safety signal</li> </ul>
<b>SF_Equivalent_REAL</b> (→ p. 388)	<ul style="list-style-type: none"> <li>compares two input values [REAL] with each other</li> <li>verifies the values with regard to the permissible value range and permissible deviation</li> <li>provides the result as a safety signal</li> </ul>

## 9.7.1 fail-safe 2-channel evaluation of the inputs

24742



The two input channels of a 2-channel design must be in different input groups to meet the safety requirements.

**Specifications** (→ following examples):

- both input channels from the same connector
- one input channel from the upper half of the connector
- the other input channel from the lower half of the connector

Not required, but recommended to simplify programming:

- both input channels of the same input type
- both input channels in the same mode

## Digital input, 2-channel: Example, connector A

24853

The example shows the pin connection with maximum possible configuration:

6 OUT000	OUT PWM-25-A	IN Frequency-4	IN000	25	44 IN0300	IN Digital-B3_2k	IN Multifunction-4	IN0120	63
7 OUT001	OUT PWM-25-B	IN Frequency-4	IN001	26	45 IN0301	IN Digital-B3_2k	IN Multifunction-4	IN0121	64
8 OUT002	OUT PWM-25-A	IN Frequency-4	IN002	27	46 IN0400	IN Resistor-4	IN Multifunction-4	IN0122	65
9 OUT003	OUT PWM-25-B	IN Frequency-4	IN003	28	47 IN0401	IN Resistor-4	IN Multifunction-4	IN0123	66
10 OUT004	OUT PWM-25-A	GND_System	GND_SYS	29	48 CAN0	CAN0 L	IN Multifunction-4	IN0220	67
11 OUT005	OUT PWM-25-B	VSS31	VSS30	30	49 CAN0	CAN0 H	IN Multifunction-4	IN0221	68
12 OUT006	OUT PWM 40-Bridge-A	OUT Supply-A3V3/0V	OUT300	31	50 CAN1	CAN1 L	IN Multifunction-4	IN0222	69
13 OUT007	OUT PWM 40-Bridge-A		OUT301	32	51 CAN1	CAN1 H	IN Multifunction-4	IN0223	70
14 OUT008	OUT PWM-40	GND_OUTVoltage-4	GND_OV4	33	52 GND_RES	GND_RES	GND_A1A	GND_A1A	71
15 OUT009	VSS31	CAN0-L	CAN0	34	53 CAN0	CAN0 L	OUT PWM-25-A	OUT0280	72
16 OUT000	OUT PWM-25-A	CAN0 H	CAN0	35	54 CAN2	CAN2 H	OUT PWM-25-A	OUT0281	73
17 OUT001	OUT PWM-25-B	IN Digital-B 3_2k	IN0800	36	55 IN0800	IN Multifunction-4	OUT PWM-25-B	OUT0282	74
18 OUT002	OUT PWM-25-A	IN Digital-B 3_2k	IN0801	37	56 IN0801	IN Multifunction-4	OUT PWM-25-A	OUT0283	75
19 OUT003	OUT PWM-25-B	IN Resistor-4	IN0900	38	57 IN0802	IN Multifunction-4	OUT PWM-25-B	OUT0284	76
20 OUT004	OUT PWM-25-A	IN Resistor-4	IN0901	39	58 IN0803	IN Multifunction-4	OUT PWM-25-A	OUT0285	77
21 OUT005	OUT PWM-25-B	IN Frequency-4	IN0500	40	59 IN0700	IN Multifunction-4	OUT PWM-25-B	OUT0286	78
22 OUT006	OUT PWM 40-Bridge-A	IN Frequency-4	IN0501	41	60 IN0701	IN Multifunction-4	OUT PWM 40-Bridge-A	OUT0287	79
23 OUT007	OUT PWM 40-Bridge-A	IN Frequency-4	IN0502	42	61 IN0702	IN Multifunction-4	OUT PWM 40-Bridge-A	OUT0288	80
24 OUT008	OUT PWM-40	IN Frequency-4	IN0503	43	62 IN0703	IN Multifunction-4	OUT PWM-40	OUT0289	81

The inputs of these groups...	...can be combined with inputs of these groups
IN00nn	IN05nn
IN01nn	IN06nn
IN01nn	IN07nn
IN02nn	IN06nn
IN02nn	IN07nn
IN03nn	IN08nn
IN04nn	IN09nn

## Digital input, 2-channel: Example, connector B

24854

The example shows the pin connection with maximum possible configuration:

6	IN1100	IN Multifunction-A	IN Digital-B 3.2k	IN1300	25	44	IN1200	IN Frequency-A	OUT PWM-25-A	OUT0400	63
7	IN1101	IN Multifunction-A	IN Digital-B 3.2k	IN1301	26	45	IN1201	IN Frequency-A	OUT PWM-25-B	OUT0401	64
8	IN1102	IN Multifunction-A	IN Digital-B 3.2k	IN1302	27	46	IN1202	IN Frequency-A	OUT PWM-25-A	OUT0402	65
9	IN1103	IN Multifunction-A	IN Digital-B 3.2k	IN1303	28	47	IN1203	IN Frequency-A	OUT PWM-25-B	OUT0403	66
10	IN1200	IN Digital-A 10k	IN Digital-A 10k	IN1400	29	48	GND_SVS	GND_System	OUT PWM-25-A	OUT0404	67
11	IN1201	IN Digital-A 10k	IN Digital-A 10k	IN1401	30	49			OUT PWM-25-B	OUT0405	68
12	IN1202	IN Digital-A 10k	IN Digital-A 10k	IN1402	31	50			OUT PWM 40-Bridge-A	OUT0406	69
13	IN1203	IN Digital-A 10k	IN Digital-A 10k	IN1403	32	51	OUT3002	OUT Voltage-A 0...10V	OUT PWM 40-Bridge-A	OUT0407	70
14	GND_ANA	GND_ANA			33	52	GND_OVA	GND_OUTVoltage-A	OUT PWM-40	OUT0408	71
15					34	53					72
16	OUT0300	OUT PWM-25-A			35	54			OUT PWM-25-A	OUT0500	73
17	OUT0301	OUT PWM-25-B	IN Multifunction-A	IN1600	36	55	IN1800	IN Digital-A 10k	OUT PWM-25-B	OUT0501	74
18	OUT0302	OUT PWM-25-A	IN Multifunction-A	IN1601	37	56	IN1801	IN Digital-A 10k	OUT PWM-25-A	OUT0502	75
19	OUT0303	OUT PWM-25-B	IN Multifunction-A	IN1602	38	57	IN1802	IN Digital-A 10k	OUT PWM-25-B	OUT0503	76
20	OUT0304	OUT PWM-25-A	IN Multifunction-A	IN1603	39	58	IN1803	IN Digital-A 10k	OUT PWM-25-A	OUT0504	77
21	OUT0305	OUT PWM-25-B	IN Digital-A 10k	IN1700	40	59	IN1900	IN Frequency-A	OUT PWM-25-B	OUT0505	78
22	OUT0306	OUT PWM 40-Bridge-A	IN Digital-A 10k	IN1701	41	60	IN1901	IN Frequency-A	OUT PWM 40-Bridge-A	OUT0506	79
23	OUT0307	OUT PWM 40-Bridge-A	IN Digital-A 10k	IN1702	42	61	IN1902	IN Frequency-A	OUT PWM 40-Bridge-A	OUT0507	80
24	OUT0308	OUT PWM-40	IN Digital-A 10k	IN1703	43	62	IN1903	IN Frequency-A	OUT PWM-40	OUT0508	81

The inputs of these groups...	...can be combined with inputs of these groups
IN10nn	IN15nn
IN11nn	IN16nn
IN12nn	IN17nn
IN12nn	IN18nn
IN14nn	IN17nn
IN14nn	IN18nn

## Analogue input, 2-channel: Example, connector A

24858

The example shows the pin connection with maximum possible configuration:

6	OUT0000	OUT PWM-25-A	IN Frequency-A	IN0000	25	44	IN0300	IN Digital-B 3.2k	IN Multifunction-A	IN0100	63
7	OUT0001	OUT PWM-25-B	IN Frequency-A	IN0001	26	45	IN0301	IN Digital-B 3.2k	IN Multifunction-A	IN0101	64
8	OUT0002	OUT PWM-25-A	IN Frequency-A	IN0002	27	46	IN0400	IN Resistor-A	IN Multifunction-A	IN0102	65
9	OUT0003	OUT PWM-25-B	IN Frequency-A	IN0003	28	47	IN0401	IN Resistor-A	IN Multifunction-A	IN0103	66
10	OUT0004	OUT PWM-25-A	GND_System	GND_SVS	29	48	CAN0	CAN0 L	IN Multifunction-A	IN0200	67
11	OUT0005	OUT PWM-25-B	V8830	V8830	30	49	CAN0	CAN0 H	IN Multifunction-A	IN0201	68
12	OUT0006	OUT PWM 40-Bridge-A	OUT Supply-A 3V/30V	OUT3000	31	50	CAN1	CAN1 L	IN Multifunction-A	IN0202	69
13	OUT0007	OUT PWM 40-Bridge-A		OUT3001	32	51	CAN1	CAN1 H	IN Multifunction-A	IN0203	70
14	OUT0008	OUT PWM-40	GND_OUTVoltage-A	GND_OVA	33	52	GND_RES	GND_RES	GND_ANA	GND_ANA	71
15	OUT0009	V8830	GND_OVA	GND_OVA	34	53	GND_L	GND_L	GND_OVA	GND_OVA	72
16	OUT0000	OUT PWM-25-A	CAN0 H	CAN0	35	54	CAN2	CAN2 H	OUT PWM-25-A	OUT0200	73
17	OUT0001	OUT PWM-25-B	IN Digital-B 3.2k	IN0800	36	55	IN0800	IN Multifunction-A	OUT PWM-25-B	OUT0201	74
18	OUT0002	OUT PWM-25-A	IN Digital-B 3.2k	IN0801	37	56	IN0801	IN Multifunction-A	OUT PWM-25-A	OUT0202	75
19	OUT0003	OUT PWM-25-B	IN Resistor-A	IN0900	38	57	IN0802	IN Multifunction-A	OUT PWM-25-B	OUT0203	76
20	OUT0004	OUT PWM-25-A	IN Resistor-A	IN0901	39	58	IN0803	IN Multifunction-A	OUT PWM-25-A	OUT0204	77
21	OUT0005	OUT PWM-25-B	IN Frequency-A	IN0500	40	59	IN0700	IN Multifunction-A	OUT PWM-25-B	OUT0205	78
22	OUT0006	OUT PWM 40-Bridge-A	IN Frequency-A	IN0501	41	60	IN0701	IN Multifunction-A	OUT PWM 40-Bridge-A	OUT0206	79
23	OUT0007	OUT PWM 40-Bridge-A	IN Frequency-A	IN0502	42	61	IN0702	IN Multifunction-A	OUT PWM 40-Bridge-A	OUT0207	80
24	OUT0008	OUT PWM-40	IN Frequency-A	IN0503	43	62	IN0703	IN Multifunction-A	OUT PWM-40	OUT0208	81

The inputs of these groups...	...can be combined with inputs of these groups
IN01nn	IN06nn
IN01nn	IN07nn
IN02nn	IN06nn
IN02nn	IN07nn

## Analogue input, 2-channel: Example, connector B

24859

The example shows the pin connection with maximum possible configuration:

6	IN1100	IN Multifunction-A	IN Digital-A 3.2k	IN1300	25	44	IN1200	IN Frequency-A	OUT PWM-25-A	OUT0400	63
7	IN1101	IN Multifunction-A	IN Digital-A 3.2k	IN1301	26	45	IN1201	IN Frequency-A	OUT PWM-25-B	OUT0401	64
8	IN1102	IN Multifunction-A	IN Digital-A 3.2k	IN1302	27	46	IN1202	IN Frequency-A	OUT PWM-25-A	OUT0402	65
9	IN1103	IN Multifunction-A	IN Digital-A 3.2k	IN1303	28	47	IN1203	IN Frequency-A	OUT PWM-25-B	OUT0403	66
10	IN1200	IN Digital-A 10k	IN Digital-A 10k	IN1400	29	48	GND_SF5	GND_System	OUT PWM-25-A	OUT0404	67
11	IN1201	IN Digital-A 10k	IN Digital-A 10k	IN1401	30	49			OUT PWM-25-B	OUT0405	68
12	IN1202	IN Digital-A 10k	IN Digital-A 10k	IN1402	31	50			OUT PWM 40-Bridge-A	OUT0406	69
13	IN1203	IN Digital-A 10k	IN Digital-A 10k	IN1403	32	51	OUT3002	OUT Voltage-A 0...10V	OUT PWM 40-Bridge-A	OUT0407	70
14	GND_ANA	GND_ANA			33	52	GND_OVA	GND_OUTVoltage-A	OUT PWM 40	OUT0408	71
15					34	53					72
16	OUT0300	OUT PWM-25-A			35	54			OUT PWM-25-A	OUT0500	73
17	OUT0301	OUT PWM-25-B	IN Multifunction-A	IN1600	36	55	IN1800	IN Digital-A 10k	OUT PWM-25-B	OUT0501	74
18	OUT0302	OUT PWM-25-A	IN Multifunction-A	IN1601	37	56	IN1801	IN Digital-A 10k	OUT PWM-25-A	OUT0502	75
19	OUT0303	OUT PWM-25-B	IN Multifunction-A	IN1602	38	57	IN1802	IN Digital-A 10k	OUT PWM-25-B	OUT0503	76
20	OUT0304	OUT PWM-25-A	IN Multifunction-A	IN1603	39	58	IN1803	IN Digital-A 10k	OUT PWM-25-A	OUT0504	77
21	OUT0305	OUT PWM-25-B	IN Digital-A 10k	IN1700	40	59	IN1900	IN Frequency-A	OUT PWM-25-B	OUT0505	78
22	OUT0306	OUT PWM 40-Bridge-A	IN Digital-A 10k	IN1701	41	60	IN1901	IN Frequency-A	OUT PWM 40-Bridge-A	OUT0506	79
23	OUT0307	OUT PWM 40-Bridge-A	IN Digital-A 10k	IN1702	42	61	IN1902	IN Frequency-A	OUT PWM 40-Bridge-A	OUT0507	80
24	OUT0308	OUT PWM 40	IN Digital-A 10k	IN1703	43	62	IN1903	IN Frequency-A	OUT PWM 40	OUT0508	81

The inputs of these groups...	...can be combined with inputs of these groups
IN11nn	IN16nn

### 9.7.2 Operation as 2-channel fail-safe digital input

25344

The safety concept for operation as **2-channel fail-safe digital input** requires the following:

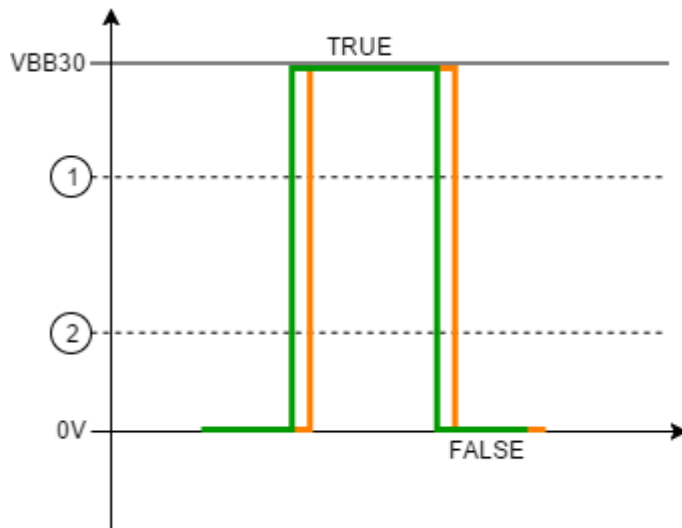
- Using 2 inputs from different input groups (→ **fail-safe 2-channel evaluation of the inputs** (→ p. 151))
- Using the **SF\_Input** (→ p. 322) in the operating mode IN\_DIGITAL\_CSI per signal
- Defining the valid signal range with the inputs uiDiagLimitMin = 0 (min value signal range) and uiDiagLimitMax = 1000 % of VBB30 (max. value signal range) on FB **SF\_Input** (→ p. 322)
- Definition of the time (detection time) as from when the deviation from the valid signal range is detected as an error at the input uiDetectionTime of the function block **SF\_Input** (→ p. 322)
- Using the function block **SF\_Equivalent** (→ p. 415) to compare both signals
- Sufficiently frequent signal change.  
In applications where this is not ensured, latent errors may occur, which can lead to a dangerous failure at the next demand upon the safety function. To minimise the risk of a latent error, it is recommended to test inputs with rarely changing signals at suitable intervals, e.g. by
  - performing a regular emergency off test
  - dynamising the signal at the start of the application
  - Using a safety switch with regular blanking pulse, → **Operation as fail-safe digital input with blanking pulses** (→ p. 146)

A suitable time interval for the execution of signal change must be derived from the safety concept for the application or the applicable product standards of the application.

**Switching thresholds (standard setting) with hysteresis for both operating modes (→ Datasheet):**

- Upper switching threshold (switching threshold\_H, FALSE -> TRUE) at 70% of VBB30

- Lower switching threshold (switching threshold\_L, TRUE -> FALSE) at 30% of VBB30



#### Legend

- |   |                       |
|---|-----------------------|
| 1 | Switching threshold_H |
| 2 | Switching threshold_L |

The following errors are detected within the diagnostic interval (→ **System architecture** (→ p. 23), → **Safety time** (→ p. 36)) is recognised:

- Exceeding of the maximum possible measuring range (→ Data sheet)

The following errors are detected when using the function block **SF\_Equivalent** (→ p. 415) within the Application Task Cycle + Discrepancy Time (→ **Application Task Cycle (APPTCY)**):

- Deviation of the two digital values for a longer time than the set time (Discrepancy Time)
  - Wire break / interruption of the signal
  - Short circuit with GND
  - Stuck-at-low (signal is stuck at FALSE)
  - Short circuit with VBB
  - Stuck-at-high (signal is stuck at TRUE)

The following errors outside the controller must be avoided or detected by means of additional measures of the application:

- Short circuit between the (sensor) cables of the 2-channel inputs (cross fault)

The function block **SF\_Equivalent** (→ p. 415) has the following behaviour if an error is detected:

- The fail-safe outputs of the function block are put into the safe state (FALSE)
- The error code can be read at the output DiagCode



→ Detailed information about the FB **SF\_Equivalent** (→ p. 415)



Only the outputs, groups and messages that are chained in the safety function will be switched off by the IEC application. Other outputs, groups and messages are not affected by this.



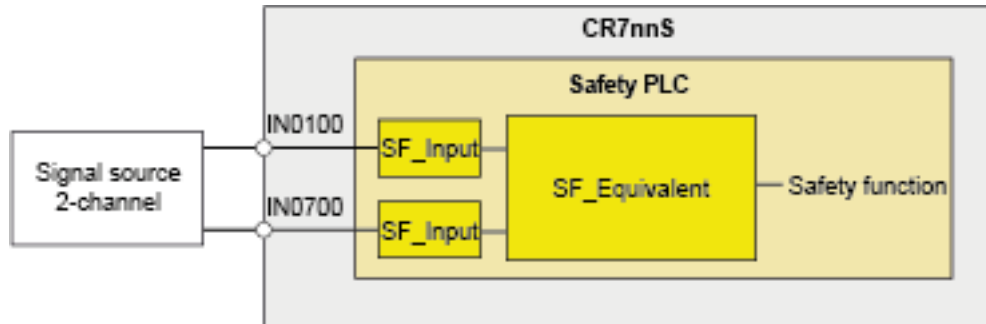
For SISTEMA:

- Use the subsystem input two-channel in the ifm VDMA library.

## Example

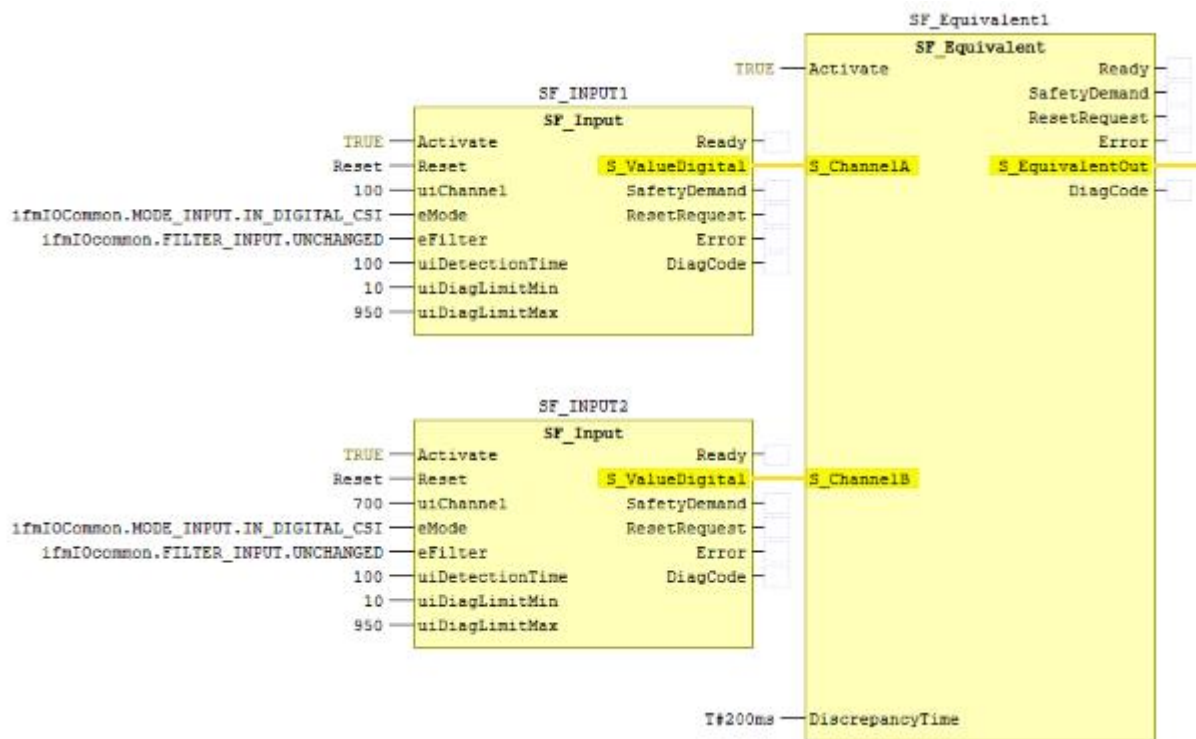
24542

Schematic illustration:



Signal source = a 2-channel, digital sensor or: two 1-channel, digital sensors

Programming example:



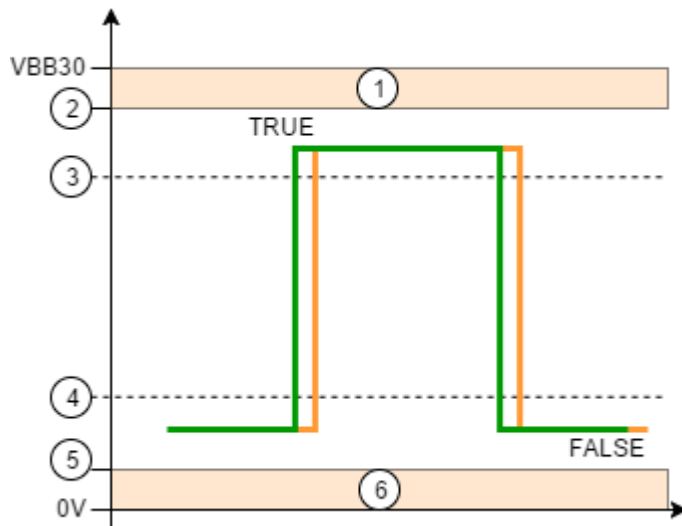
## Detailed error evaluation

25186

For faster error evaluation, the digital input signal can be restricted by means of suitable application-specific limitation of the signal range (setting the minimum and the maximum values of the signal range on the function block **SF\_Input** (→ p. 322)). Then, also the following errors can be detected within the time (detection time) set on the function block **SF\_Input** (→ p. 322):

- Short-circuit to the supply voltage VBB30
- Short circuit to GND

- Wire break / interruption of the signal



#### Legend

- |   |   |
|---|---|
| 1 | Short-circuit to VBB30  |
| 2 | Signal range maximum value                                      |
| 3 | Switching threshold_H   |
| 4 | Switching threshold_L   |
| 5 | Signal range minimum value                                      |
| 6 | Short-circuit to GND or wire break / interruption of the signal |

As an option, it is possible to use the operating mode IN\_DIGITAL\_CSI\_NAMUR with the fixed values:

Mode	Min. value	Max. value
IN_DIGITAL_CSI_NAMUR	1 V	95% of VBB30

### 9.7.3 Operation as 2-channel fail-safe analogue input

58734

The safety concept for IN\_MULTIFUNCTION\_A for operation as **2-channel fail-safe analogue input** requires the following:

- Using 2 inputs from different input groups (→ **fail-safe 2-channel evaluation of the inputs** (→ p. 151))
- Using the FB **SF\_Input** (→ p. 322) in the operating mode per signal:
  - IN\_CURRENT\_CSI
  - IN\_VOLTAGE\_10
  - IN\_VOLTAGE\_32
  - IN\_VOLTAGE\_RATIO
- Detection of errors through the deviation between the 2 signals
- Determination of the admissible deviation of the 2 signals on the input AcceptTolerance (absolute value) of the FB **SF\_Equivalent\_DINT** (→ p. 379), **SF\_Equivalent\_UINT** (→ p. 382), **SF\_Equivalent\_UDINT** (→ p. 385) or **SF\_Equivalent\_REAL** (→ p. 388).
- Sufficiently frequent signal change.  
In applications where this is not ensured, latent errors may occur, which can lead to a dangerous



failure at the next demand upon the safety function. To minimise the risk of a latent error, it is recommended to test inputs with rarely changing signals at suitable intervals, e.g. by

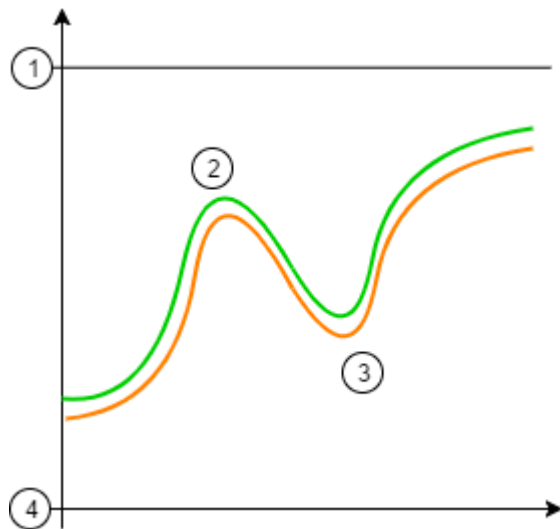
- dynamising the signal at the start of the application

A suitable time interval for the execution of signal change must be derived from the safety concept for the application or the applicable product standards of the application.

**Signal range for the individual modes:** → Data sheet

**The following errors are detected within the diagnostic test interval ( → **Diagnostic Test Intervall (DTI)** (→ p. 32)):**

- Exceeding of the maximum possible measuring range ( → Data sheet)



#### Legend

- |   |                               |
|---|-------------------------------|
| 1 | Measuring value maximum range |
| 2 | Analogue signal 1st channel   |
| 3 | Analogue signal 2nd channel   |
| 4 | 0 [V / mA / %]                |

**The following errors are detected when using the FB **SF\_Equivalent\_DINT** (→ p. 379), **SF\_Equivalent\_UINT** (→ p. 382), **SF\_Equivalent\_UDINT** (→ p. 385) or **SF\_Equivalent\_REAL** (→ p. 388) within the Application Task Cycle ( → **Time-related behaviour IEC application** (→ p. 33)):**

- if the 2 signals deviate for more than the set admissible value on AcceptTolerance
  - Wire break / interruption of the signal
  - Short circuit with GND
  - Stuck-at-Low (signal is stuck at 0 mV / mA)
  - Short circuit with VBB
  - Stuck-at (signal is stuck at one static analogue value)

**The following errors outside the controller must be avoided or detected by means of additional measures of the application:**

- Short circuit between the (sensor) cables of the 2-channel inputs (cross fault)

**The function blok **SF\_Equivalent\_DINT** (→ p. 379), **SF\_Equivalent\_UINT** (→ p. 382), **SF\_Equivalent\_UDINT** (→ p. 385) or **SF\_Equivalent\_REAL** (→ p. 388) has the following behaviour if an error is detected:**

- The fail-safe outputs of the function block are put into the safe state (0)
- The error code can be read at the output DiagCode

→ Detailed information about:



- FB **SF\_Equivalent\_DINT** (→ p. [379](#))
- FB **SF\_Equivalent\_UINT** (→ p. [382](#))
- FB **SF\_Equivalent\_UDINT** (→ p. [385](#))
- FB **SF\_Equivalent\_REAL** (→ p. [388](#))



Only the outputs, groups and messages that are chained in the safety function will be switched off by the IEC application. Other outputs, groups and messages are not affected by this.



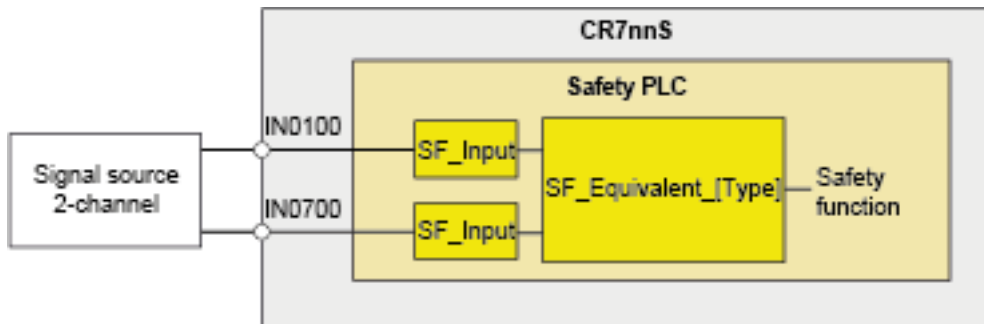
For SISTEMA:

- Use the subsystem input `two-channel` in the ifm VDMA library.

## Example

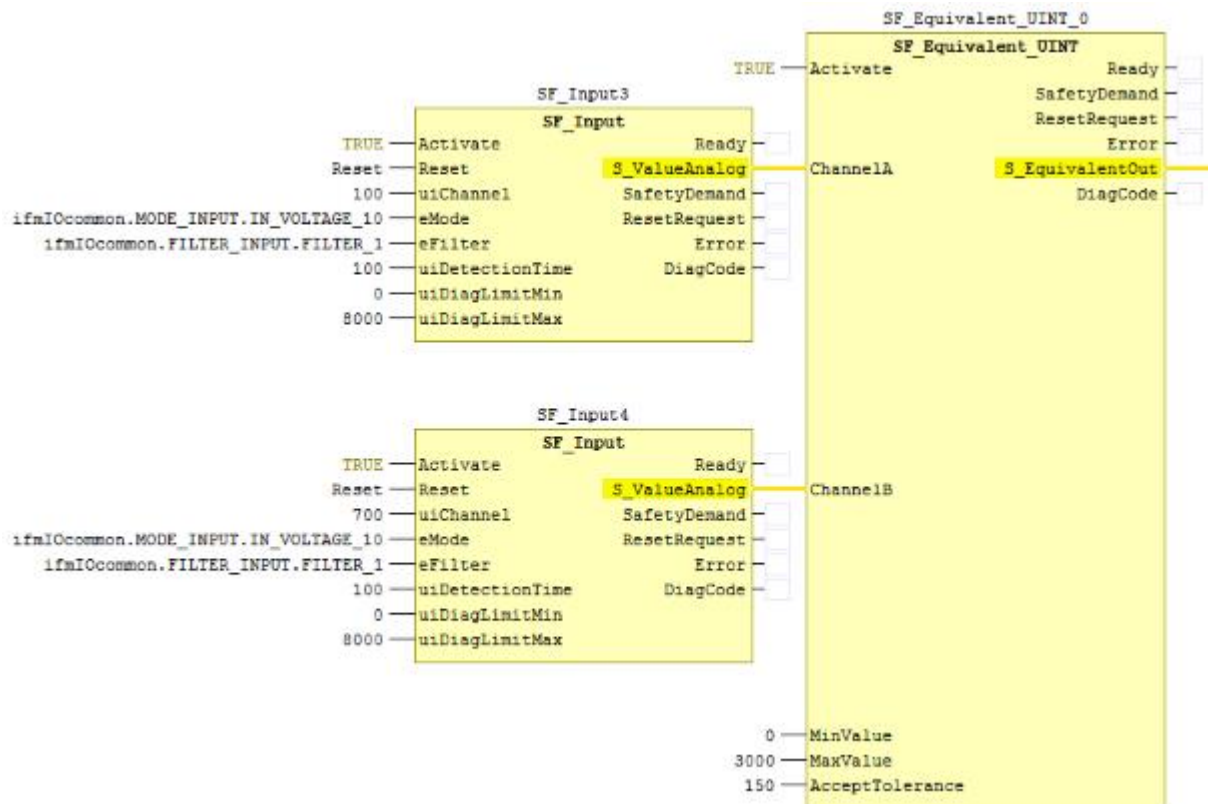
24751

Schematic illustration:



Signal source = one 2-channel, analogue sensor or: two 1-channel, analogue sensors  
 [Type] = REAL or UDINT or UINT or DINT

Programming example SF\_Equivalent\_UINT:

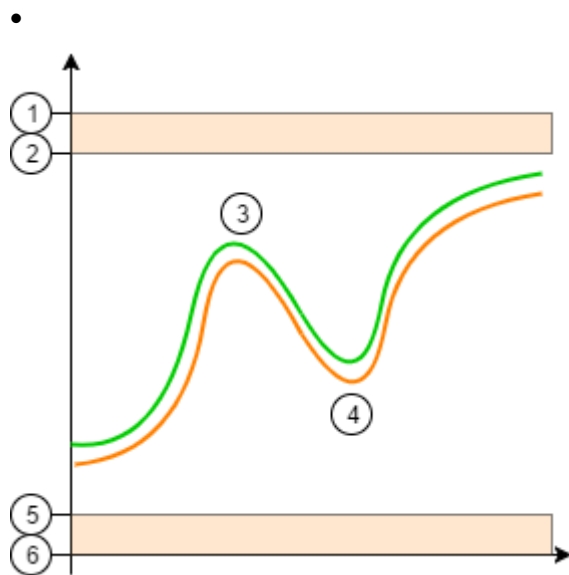


## Detailed error evaluation

25349

To evaluate errors in greater detail, the analogue input signal can be restricted by means of suitable application-specific limitation of the signal range (setting the minimum and the maximum values of the signal range on the function block **SF\_Input** (→ p. 322)). Then, also the following errors can be detected within the time (detection time) set on function block **SF\_Input** (→ p. 322):

- Short-circuit to the supply voltage VBB30
- Short circuit to GND
- Wire break / interruption of the signal



#### Legend

- 1 Measuring value maximum range
- 2 Signal range maximum value
- 3 Analogue signal 1st channel
- 4 Analogue signal 2nd channel
- 5 Signal range minimum value
- 6 0 [V / mA / %]

### 9.7.4 Operation as 2-channel fail-safe frequency input

60524

The safety concept for IN\_FREQUENCY\_B for operation as **2-channel fail-safe frequency input** requires the following:

- Using 2 inputs from different input groups (→ **fail-safe 2-channel evaluation of the inputs** (→ p. 151))
- Using the following operating modes:
  - IN\_FREQUENCY\_CSI
  - IN\_INC\_ENCODER\_CSI
  - IN\_PERIOD\_RATIO\_CSI
  - IN\_PERIOD\_RATIO\_US\_CSI
  - IN\_PHASE\_CSI
  - IN\_COUNT\_CSI
- Detection of errors through the deviation between the 2 signals
- Determination of the admissible deviation of the 2 signals on the input AcceptTolerance (absolute value) of the FB **SF\_Equivalent\_DINT** (→ p. 379), **SF\_Equivalent\_UINT** (→ p. 382), **SF\_Equivalent\_UDINT** (→ p. 385) or **SF\_Equivalent\_REAL** (→ p. 388).

**Signal range for the individual modes:** → Data sheet.

**The following errors are displayed when using the FB **SF\_Equivalent\_DINT** (→ p. 379), **SF\_Equivalent\_UINT** (→ p. 382), **SF\_Equivalent\_UDINT** (→ p. 385) or **SF\_Equivalent\_REAL** (→ p. 388) within the Application Task Cycle (→ **Time-related behaviour IEC application** (→ p. 33)):**

- if the 2 signals deviate for more than the set admissible value on AcceptTolerance

- Wire break / interruption of the signal
- Short circuit with GND
- Stuck-at-Low (signal is stuck at 0 mV / mA)
- Short circuit with VBB
- Stuck-at-high (signal is stuck at TRUE)

**The following errors outside the controller must be avoided or detected by means of additional measures of the application:**

- Short circuit between the (sensor) cables of the 2-channel inputs (cross fault)

**The function block `SF_Equivalent_DINT` (→ p. [379](#)), `SF_Equivalent_UINT` (→ p. [382](#)), `SF_Equivalent_UDINT` (→ p. [385](#)) or `SF_Equivalent_REAL` (→ p. [388](#)) has the following behaviour if an error is detected:**

- The fail-safe outputs of the function block are put into the safe state (0)
- The error code can be read at the output `DiagCode`

→ Detailed information about:



- FB `SF_Equivalent_DINT` (→ p. [379](#))
- FB `SF_Equivalent_UINT` (→ p. [382](#))
- FB `SF_Equivalent_UDINT` (→ p. [385](#))
- FB `SF_Equivalent_REAL` (→ p. [388](#))



Only the outputs, groups and messages that are chained in the safety function will be switched off by the IEC application. Other outputs, groups and messages are not affected by this.



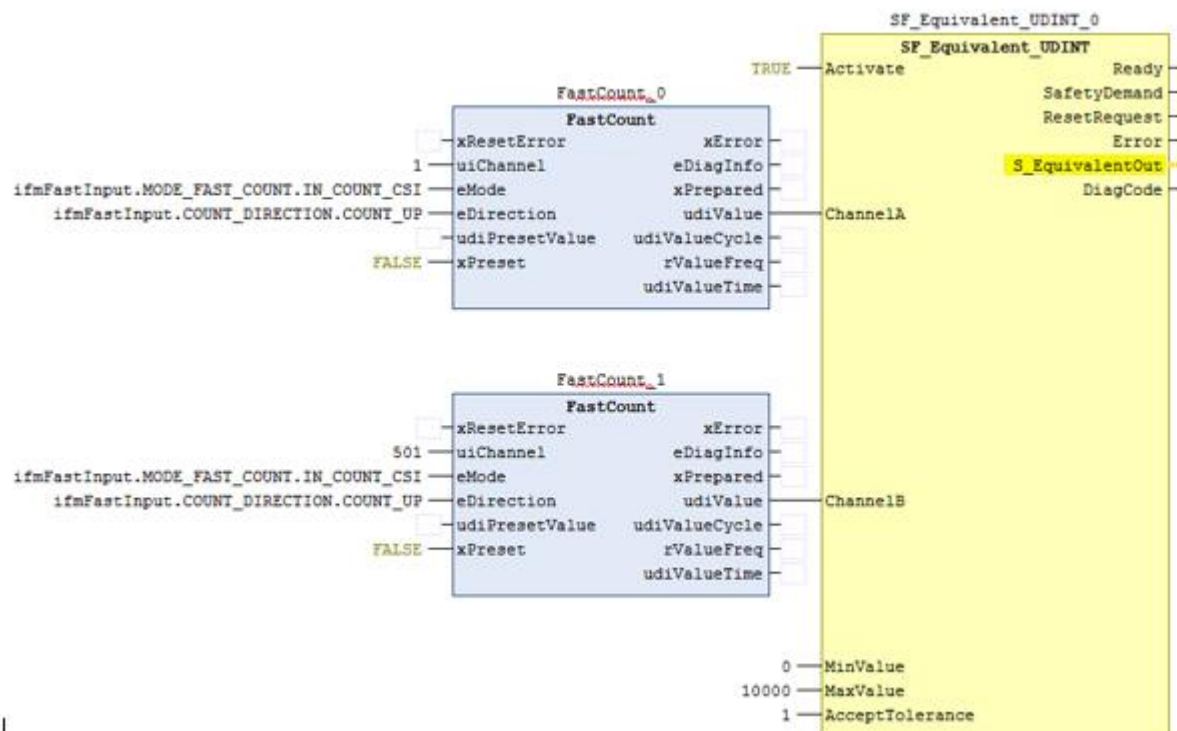
For SISTEMA:

- Use the subsystem input `two-channel` in the `ifm VDMA` library.

## Example

24857

Programming example SF\_Equivalent\_UDINT during frequency operation:



## 9.8 Safety concept of the outputs

### Content

Operation as fail-safe digital output .....	164
Operation as PWM output and current-controlled output .....	166
Output, 1-channel, safe .....	168
Output 2-channel, safe, with output group .....	169
Output, 2-channel, safe, without output group .....	170
Switching off an output group safely .....	172

24550

Depending on the requirement, a safety-related output can have the following design:

- 1-channel
- 1-channel with function monitoring
- 2-channel
- 2-channel with function monitoring

This is admissible with the following output types:

Output type	permissible operating mode (eMode) for safety operation	function block required for safety operation
OUT PWM-25-A	OUT_DIGITAL_CSO OUT_PWM_CSO OUT_CURRENT_CSO	<b>SF_OutputEnh</b> (→ p. 332) <b>SF_PWM1000Enh</b> (→ p. 341) <b>SF_CurrentControlEnh</b> (→ p. 346)
OUT PWM-25-B	OUT_DIGITAL_CSO OUT_PWM_CSO	<b>SF_OutputEnh</b> (→ p. 332) <b>SF_PWM1000Enh</b> (→ p. 341)
OUT PWM-40-A	OUT_DIGITAL_CSO OUT_PWM_CSO OUT_CURRENT_CSO	<b>SF_OutputEnh</b> (→ p. 332) <b>SF_PWM1000Enh</b> (→ p. 341) <b>SF_CurrentControlEnh</b> (→ p. 346)
OUT PWM-40-BRIDGE-A	OUT_DIGITAL_CSO OUT_PWM_CSO OUT_CURRENT_CSO	<b>SF_OutputEnh</b> (→ p. 332) <b>SF_PWM1000Enh</b> (→ p. 341) <b>SF_CurrentControlEnh</b> (→ p. 346)

For the following output type, the use as fail-safe H-bridge circuit is admissible:

Output type	function block required for safety operation
OUT PWM-40-BRIDGE-A	<b>SF_HBridgeEnh</b> (→ p. 351)

\*with standard value (eBrakeMode=FALSE)

Depending on the requirement, an entire output group can be used in a safety-related context:

permissible operating mode (eMode) for safety operation	function block required for safety operation
OUT_DIGITAL	<b>SF_OutGroupEnh</b> (→ p. 336)

24963



► Please consider when setting up the safety function:

The 1-channel design (with or without function monitoring) requires protected wiring from the output terminal to the actuator!

To access to the fail-safe outputs of the device, the following safety function blocks are available:

Function element	Short description
<b>SF_OutputEnh</b> (→ p. 332)	Assigns an operating mode to a one-channel fail-safe output channel and provides the current state at the selected channel. Output deactivation is fail-safe.
<b>SF_OutGroupEnh</b> (→ p. 336)	Controller of an output group with fail-safe deactivation.
<b>SF_CurrentControlEnh</b> (→ p. 346)	Controller of a current controlled output with fail-safe deactivation.
<b>SF_PWM1000Enh</b> (→ p. 341)	Controller of a PWM output with fail-safe deactivation.

Function element	Short description
<b>SF_HBridgeEnh</b> (→ p. <a href="#">351</a> )	Controlling an output channel pair with the H-bridge operating mode with fail-safe deactivation.



- In new applications, use the enhanced function blocks (SF\_[Type]\_Enh) described in the table.

The previous function modules can continue to be used in existing applications for reasons of compatibility.

## 9.8.1 Operation as fail-safe digital output

58736



→ Detailed information about the FB **SF\_OutputEnh** (→ p. [332](#))

The safety concept for operation as **fail-safe digital output** requires the following:

- Using the function block SF\_OutputEnh in the operating mode OUT\_DIGITAL\_CS0
- The output monitors the maximum admissible output current in accordance with the output type.  
→ Datasheet.
- Successful start-up test of the group switch including the outputs
- Sufficiently frequent signal change  
In applications where this is not ensured, latent errors may occur, which can lead to a dangerous failure at the next demand upon the safety function. To minimise the risk of a latent error, it is recommended to test the group switch including the outputs with rarely changing signals at suitable intervals, e.g. by:
  - Restart of the controller via PowerOn reset or the FB reset

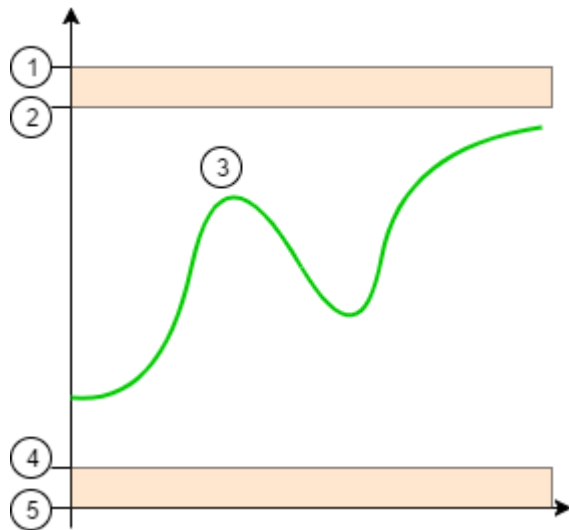
A suitable time interval for the execution of signal change must be derived from the safety concept for the application or the applicable product standards of the application.

- Depending on the application, set the following on SF\_OutputEnh:

- The valid current range at the inputs uiDiagLimitMin and uiDiagLimitMax.
- Determination of the time delay from when the departure from the valid current signal is detected as an error at the input uiDetectionTime.



### Curve of the output current (example)



#### Legend

- |   |   |
|---|---|
| 1 | Maximum admissible output current   |
| 2 | Maximum value of the current range for excessive current diagnostics                |
| 3 | Output current  |
| 4 | Minimum value of the current range for wire break detection (standard value = 0 mA) |
| 5 | 0 mA  |

The following **errors within the safety controller** are detected when using the function block SF\_OutputEnh if the error has been present for at least as long as the set `uiDetectionTime` :

- Stuck-at-Low (signal is stuck on FALSE) while the output is switched on
- Stuck-at-High (signal is stuck on TRUE) before the application is started and each time the output is switched off
- Excessive current
- Short circuit to GND (through excessive current diagnostics)

The **errors outside the controller** must be prevented or lead to the safe state by taking additional measures of the application:

- Short circuit to other live leads outside the output group
- Short circuit to supply voltages VBBnn

The project engineer can adapt the current range of the output specifically to the application. The following **errors** can then be detected if the error is present for at least as long as the set `uiDetectionTime` :

- Wire break
- Irregular current consumption of the load

The output has the following behaviour if an error is detected:

- The safe output is put into the safe state (0 and FALSE)
- In case of Stuck-at-High, the corresponding output group is switched off
- The error code can be read at the output `DiagCode` of the SF\_OutputEnh



Only the affected output or the corresponding output group including the corresponding outputs will be switched off (in case of Stuck-at-High). Other output groups, inputs and messages are not affected by this.

## 9.8.2 Operation as PWM output and current-controlled output

58737



→ Detailed information about the FB **SF\_PWM1000Enh** (→ p. [341](#))

→ Detailed information about the FB **SF\_CurrentControlEnh** (→ p. [346](#))

The safety concept for outputs during operation as PWM output requires the following:

- Using the function block SF\_PWM1000Enh in the operating mode OUT\_PWM\_CSO.
- The output monitors the maximum admissible output current in accordance with the output type.  
→ Datasheet.

The safety concept for outputs during operation as **current-controlled output** provides the following:

- Using the function block SF\_CurrentControl in the operating mode OUT\_CURRENT\_CSO.
- The output monitors the maximum admissible output current in accordance with the output type.  
→ Datasheet.

The safety concept for outputs in both operating modes requires the following:

- Sufficiently frequent signal change  
In applications where this is not ensured, latent errors may occur, which can lead to a dangerous failure at the next demand upon the safety function. To minimise the risk of a latent error, it is recommended to test the group switch including the outputs with rarely changing signals at suitable intervals, e.g. by:
  - Restart of the controller via PowerOn reset or the FB reset

A suitable time interval for the execution of signal change must be derived from the safety concept for the application or the applicable product standards of the application.

Depending on the application, set the following:

- The valid current range at the inputs uiDiagLimitMin and uiDiagLimitMax.
- Determination of the time delay from when the departure from the valid current signal is detected as an error at the input uiDetectionTime.

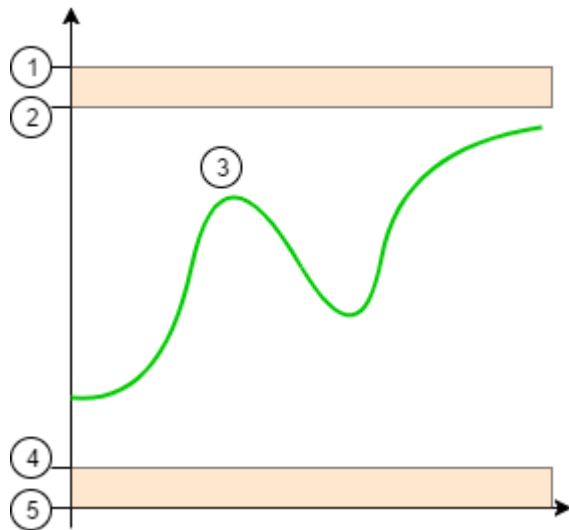


In OUT\_PWM\_CSO, OUT\_CURRENT\_CSO and OUT\_H\_BRIDGE modes, the uiDetectionTime must be at least twice the PWM frequency. If dithering is used, 2 times the dither frequency must be used.

Example:

Frequency = 100 Hz

$uiDetectionTime(min) = 1 / 100 \text{ Hz} * 2 = 20 \text{ ms}$

**Curve of the output current (example)****Legend**

- |   |   |
|---|---|
| 1 | Maximum admissible output current   |
| 2 | Maximum value of the current range for excessive current diagnostics                |
| 3 | Output current  |
| 4 | Minimum value of the current range for wire break detection (standard value = 0 mA) |
| 5 | 0 mA  |

The following **errors within the safety controller** are detected when using the FB SF\_PWM1000Enh or FB SF\_CurrentControlEnh, if the error is present for at least as long as the set uiDetectionTime:

- Stuck-at-High (signal is stuck on TRUE) before the application is started and each time the output is switched off
- Excessive current
- Short circuit to GND (through excessive current diagnostics)

The **errors outside the controller** must be prevented or lead to the safe state by taking additional measures of the application:

- Short circuit to other live leads outside the output group
- Short circuit to supply voltages VBBnn

The project engineer can adapt the current range of the output specifically to the application. The following **errors** can then be detected if the error is present for at least as long as the set uiDetectionTime :

- Wire break
- Stuck-at-Low (signal is stuck on FALSE) is only possible above the current range
- Irregular current consumption of the load

The output has the following behaviour if an error is detected:

- The safe output is put into the safe state (0 and FALSE)
- In case of Stuck-at-High, the corresponding output group is switched off
- The error code can be read at the output DiagCode of the function block SF\_PWM1000Enh or FB SF\_CurrentControl



Only the affected output or the corresponding output group including the corresponding outputs will be switched off (in case of Stuck-at-High). Other output groups, inputs and messages are not affected by this.

### 9.8.3 Output, 1-channel, safe

58738

The safety concept for one **1-channel output** requires the following:

- Safe switch-off of a 1-channel output
- Sufficiently frequent signal change  
In applications where this is not ensured, latent errors may occur, which can lead to a dangerous failure at the next demand upon the safety function. To minimise the risk of a latent error, it is recommended to test the group switch including the outputs with rarely changing signals at suitable intervals, e.g. by:
  - Restart of the controller via PowerOn reset or the FB reset
- A suitable time interval for the execution of signal change must be derived from the safety concept for the application or the applicable product standards of the application.
- Output with use of the output group switch as second option for safety shut-down

24963



► Please consider when setting up the safety function:

The 1-channel design (with or without function monitoring) requires protected wiring from the output terminal to the actuator!



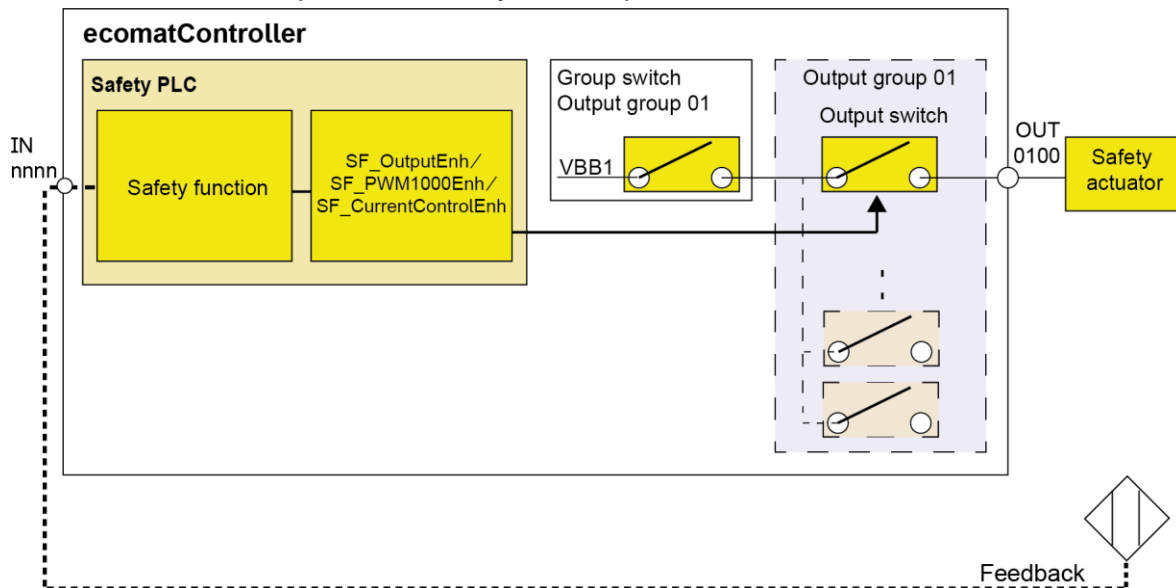
For SISTEMA:

► Use the subsystem output `one-channel` in the ifm VDMA library.

## Example

54459

### Schematic illustration (1-channel safety actuator):



When requesting the safety function, the controller switches the output switch off.

> The corresponding output switches off.

The group switch has no significance for the safety function. Option for shut-down if the output switch has a stuck-at-high error (cannot be switched off)

As an option and to improve the diagnostics in the application, a feedback of the mechanical safety function can be implemented (e.g. sensors, encoders, limit switches).

## 9.8.4 Output 2-channel, safe, with output group

58739

The safety concept for one 2-channel output requires the following:

- Safe switch-off of a 2-channel output
- Use of the output group switch as second option for safety shut-down
- Sufficiently frequent signal change
  - In applications where this is not ensured, latent errors may occur, which can lead to a dangerous failure at the next demand upon the safety function. To minimise the risk of a latent error, it is recommended to test group switches including the outputs with rarely changing signals at suitable intervals, e.g. by:
    - Restart of the controller via PowerOn reset or the FB reset
- A suitable time interval for the execution of signal change must be derived from the safety concept for the application or the applicable product standards of the application.



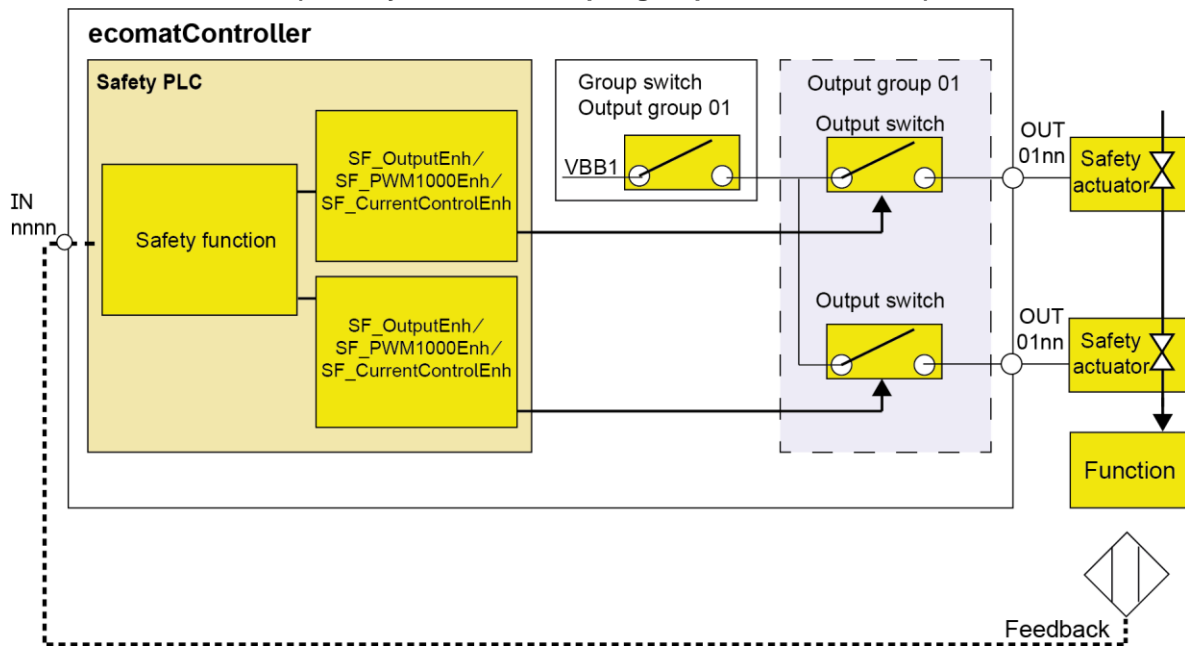
Depending on the type of outputs, please note the following paragraphs:

- → **Operation as fail-safe digital output** (→ p. [164](#))
- → **Operation as PWM output and current-controlled output** (→ p. [166](#))

## Example

58740

### Schematic illustration (2 safety actuators, output group switch activated):



When requesting the safety function, the controller deactivates both output switches in parallel.

> The corresponding outputs switch off.

For the safety function, the group switch has the significance of second option for safety shut-down when an output switch has a Stuck-at-High error (cannot be switched off).

As an option and to improve the diagnostics in the application, a feedback of the mechanical safety function can be implemented (e.g. sensors, encoders, limit switches).



For SISTEMA:

- Use the subsystem output two-channel, outputs in the same output group in the ifm VDMA library.

## 9.8.5 Output, 2-channel, safe, without output group

58741

The safety concept for one **2-channel output** requires the following:

- Safe switch-off of a 2-channel output without using the output group switch
- Sufficiently frequent signal change  
In applications where this is not ensured, latent errors may occur, which can lead to a dangerous failure at the next demand upon the safety function. To minimise the risk of a latent error, it is recommended to test group switches including the outputs with rarely changing signals at suitable intervals, e.g. by:
  - Restart of the controller via PowerOn reset or the FB reset
- A suitable time interval for the execution of signal change must be derived from the safety concept for the application or the applicable product standards of the application.



For 2-channel layout without using the output group switch:

- Deactivation of the output group switch with the function block **ConfigDiagProt** (→ p. [294](#)) in the operating mode OUT\_DIGITAL as long as the latter is not used by other outputs of the same group as safety function.
- The following combination of outputs of these groups are permitted:

The outputs of these groups...	...may be combined with outputs of these groups
OUT00nn	OUT01nn
OUT00nn	OUT04nn
OUT01nn	OUT02nn
OUT01nn	OUT04nn
OUT02nn	OUT03nn
OUT03nn	OUT04nn



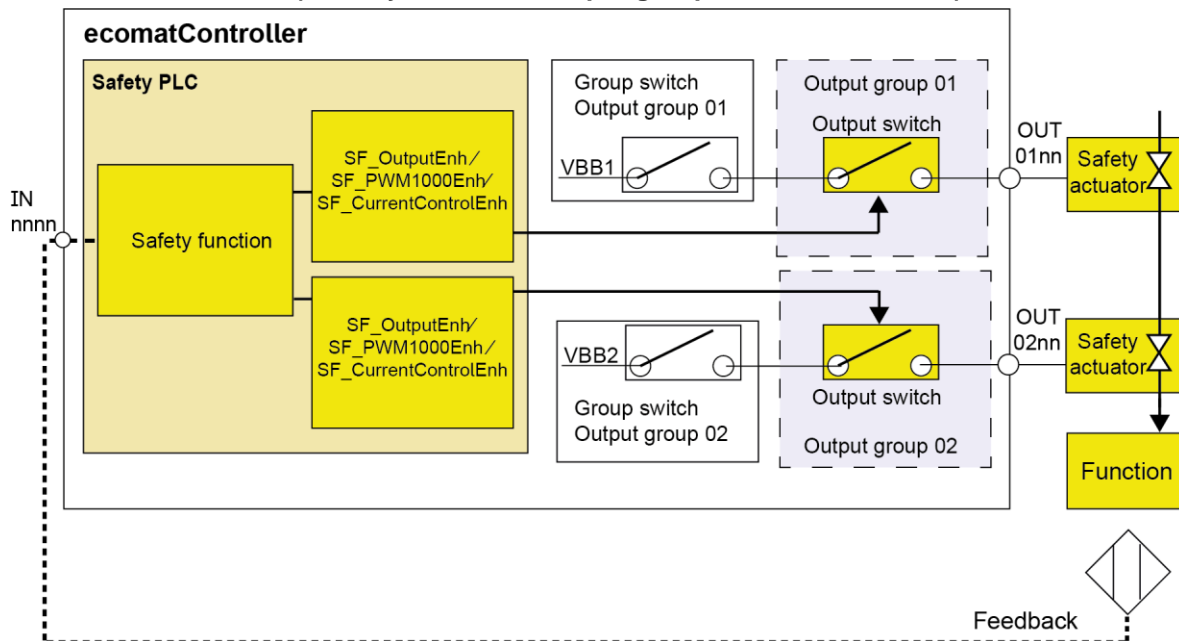
Depending on the type of outputs, please note the following paragraphs:

- → **Operation as fail-safe digital output** (→ p. [164](#))
- → **Operation as PWM output and current-controlled output** (→ p. [166](#))

## Example

58742

### Schematic illustration (2 safety actuators, output group switch deactivated):



When requesting the safety function, the controller switches the output switches off.

> The corresponding outputs switch off.

The group switch has no significance for the safety function because it is deactivated.

As an option and to improve the diagnostics in the application, a feedback of the mechanical safety function can be implemented (e.g. sensors, encoders, limit switches).



For SISTEMA:

- Use the subsystem output two-channel, outputs in different output groups in the ifm VDMA library.

## 9.8.6 Switching off an output group safely

58743



→ Detailed information about the FB **SF\_OutGroupEnh** (→ p. 336)

The safety concept for one **output group** requires the following:

- Using the function block **SF\_OutGroupEnh** (→ p. 336) in the OUT\_DIGITAL operating mode
- The output group monitors the maximum admissible total current of all corresponding outputs.  
→ Data sheet
- When switched off, the group switch and the corresponding outputs will be switched off by the safe ifm operating system irrespective of the requirements of the IEC application.
- An error on one switch will not cause the loss of the safety function
- Sufficiently frequent signal changes at the output group  
In applications where this is not ensured, latent errors may occur, which can lead to a dangerous failure at the next demand upon the safety function. To minimise the risk of a latent error, it is recommended to test the group switch at suitable intervals, e.g. by:

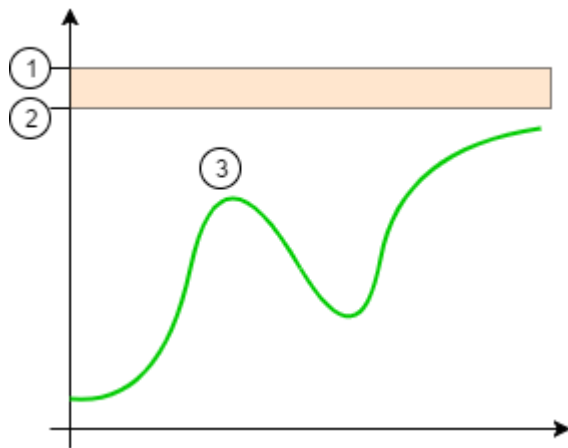


- Restart of the controller via PowerOn reset or the FB reset
- Switch-off of the output group with the FB **SF\_OutGroupEnh** (→ p. 336)
- A suitable time interval for the execution of signal change must be derived from the safety concept for the application or the applicable product standards of the application.

Depending on the application, set the following on SF\_OutGroupEnh:

- Always set the uiDiagLimitMin input to 0.
- The valid maximum current value at input uiDiagLimitMax.
- Determination of the time delay from when the departure from the valid current signal is detected as an error at the input uiDetectionTime.

#### Curve of the total current of the output group (example)



#### Legend

- |   |  |
|---|--|
| 1 | Maximum admissible output current                                    |
| 2 | Maximum value of the current range for excessive current diagnostics |
| 3 | Total current of the output group                                    |

The following **faults within the safety controller** are detected when using the FB **SF\_OutGroupEnh** (→ p. 336) if the error lasts at least as long as the set uiDetectionTime:

- Stuck-at-Low (signal is stuck on FALSE) while the output group is switched on
- Stuck-at-High (signal is stuck at TRUE) before the application is started and each time the output group is switched off
- Excessive current of the total current of the corresponding outputs

The **errors outside the controller** must be prevented or lead to the safe state by taking additional measures of the application:

- Short circuit to other live leads outside the output group
- Short circuit to supply voltages VBBnn

The project engineer can adapt the maximum current value of the output group to the specific application. The following **errors** can then be detected if the error is present for at least as long as the set uiDetectionTime :

- Irregular total current of the corresponding output loads

The output group and the corresponding outputs have the following behaviour if an error is detected:

- The group switch and the corresponding outputs are synchronously put into the safe state (0 and FALSE)
- The error code can be read at the output DiagCode of the SF\_OutGroupEnh



Only the affected output group will be shut down. Other output groups, inputs and messages are not affected by this.



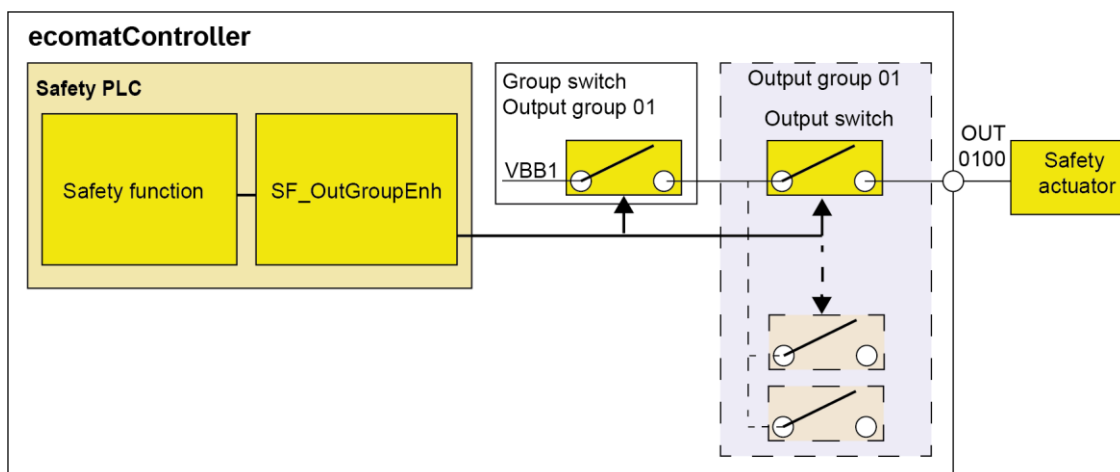
For SISTEMA:

- Use the subsystem output one-channel in the ifm VDMA library.

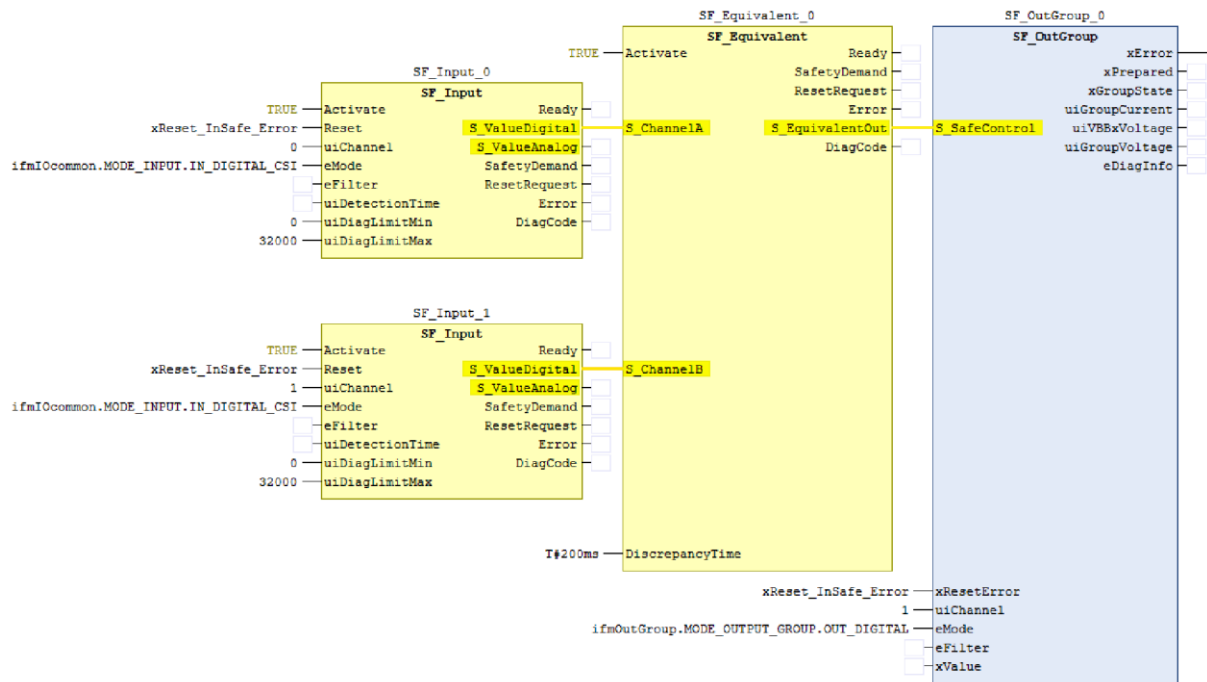
## Example

25424

### Schematic illustration:



If the group is switched off by the function block **SF\_OutGroupEnh** (→ p. 336), all outputs of the group will likewise be automatically switched off safely, even if one of the outputs is used by the safety PLC or the standard PLC.

**Programming example:**

The example is a 2-channel E-stop function with fail-safe shut-down of the output group.

## 9.9 Use CANopen-Safety

### Content

CANopen safety error behaviour .....	176
Characteristic safety values for CANopen Safety .....	177

24547



► Observe the notes on task configuration! (→ **Configure task processing** (→ p. [86](#)))

To access a CAN interface configured for CANopen safety operation in a safety application, POU's are available in CODESYS libraries by CODESYS GmbH.

#### Requirements

- The device is configured as CANopen manager SIL2 (master)  
→ **via system configuration: CANopen manager SIL2** (→ p. [100](#))
- A CANopen-Device SIL2 (slave) is configured  
→ **via system configuration: CANopen-Device SIL2** (→ p. [101](#))

If the service communication and the process communication are simultaneously operated on the same CAN interface, the project engineer must ensure that the process communication of the safety application will not be influenced (e.g. choose a CAN ID with a lower priority for service communication).

If the process communication of both IEC applications are simultaneously operated on the same CAN interface, the project engineer must ensure that the process communication of the safety application will not be influenced (e.g. choose a CAN ID with a lower priority for standard process communication).

### 9.9.1 CANopen safety error behaviour

25358

The CODESYS Ctrl SIL2 implements the safety mechanisms according to EN50325-5, table 1 as follows:

The following errors will be detected:

- Corruption
- Unintended Repetition
- Incorrect Sequence
- Loss
- Unacceptable delay
- Insertion

The following reaction occurs if an error is detected:

- communication to the affected slave will stop
- at the function block, the output `S_[Safety Slave Name].S_xActive = FALSE` will be set
- at the function block, an error code will be indicated at the output `S_[Safety Slave Name].S_eError` :

Communication error	Error code
Corruption	SRDO_DATA_ERROR
Unintended Repetition	SRDO_RECEIVE_ERROR

Incorrect Sequence	SRVT_TIMEOUT
Loss	SCT_TIMEOUT
Unacceptable delay	SCT_TIMEOUT
Insertion	Depending in the type and the time behaviour: <ul style="list-style-type: none"> <li>• SRDO_RECEIVE_ERROR</li> </ul> or <ul style="list-style-type: none"> <li>• SRVT_TIMEOUT</li> </ul>

25425



## WARNING

In case of a fault, the value of the SRDO will stay on the last value before the error occurred.

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function.
- ▶ The output `S_[Safety Slave Name].S_xActive` must be polled for TRUE before using the SRDOs.
- ▶ For the case that the output `S_[Safety Slave Name].S_xActive = FALSE`, take measures to handle errors in the application.

The following errors will NOT be detected and must be investigated by the user:

- Masquerade
- Addressing

The user is responsible for the correctness of the configuration. In this context, the user must not only consider the fail-safe I/Os, but the entire bus configuration within and outside the user's CANopen configuration in CODESYS.

Here, it is particularly important to check the following possible wrong configurations:

- Correctness and non-overlap of the PDO mappings
- Correctness and non-overlap of the SRDO mappings
- Overlaps of CobIDs
- Overlaps of NodeIDs
- Different baud rates of the participants



- ▶ Use the button [Konfiguration prüfen und korrigieren] in the CANopen Manager SIL2 > [Allgemein] in CODESYS to check the wrong configuration.

→ [H2] User manual CODESYS Safety SIL 2 V6.0 of CODESYS GmbH, §H2-6.4 CANopen Safety Stack: Review of the configuration

## 9.9.2 Characteristic safety values for CANopen Safety

58744

To simplify the application of characteristic safety values for CANopen Safety, the following basic conditions apply for each safety function:

- Subsystem CANopen Safety: 1% of the permitted PFH<sub>D</sub> (for SIL2 < 1.0 x 10<sup>-8</sup>)
- A maximum of 5000 safety-related messages (SRDOs) per second

For simplified representation the following applies based on the specifications in IEC 61784-3 and DIN EN 50325-5:

- ▶ Use PFH<sub>D</sub> value  $1.0 \times 10^{-8}$ .
- ▶ In SISTEMA: Use a "CANopen Safety" subsystem with PFH<sub>D</sub> =  $1.0 \times 10^{-8}$



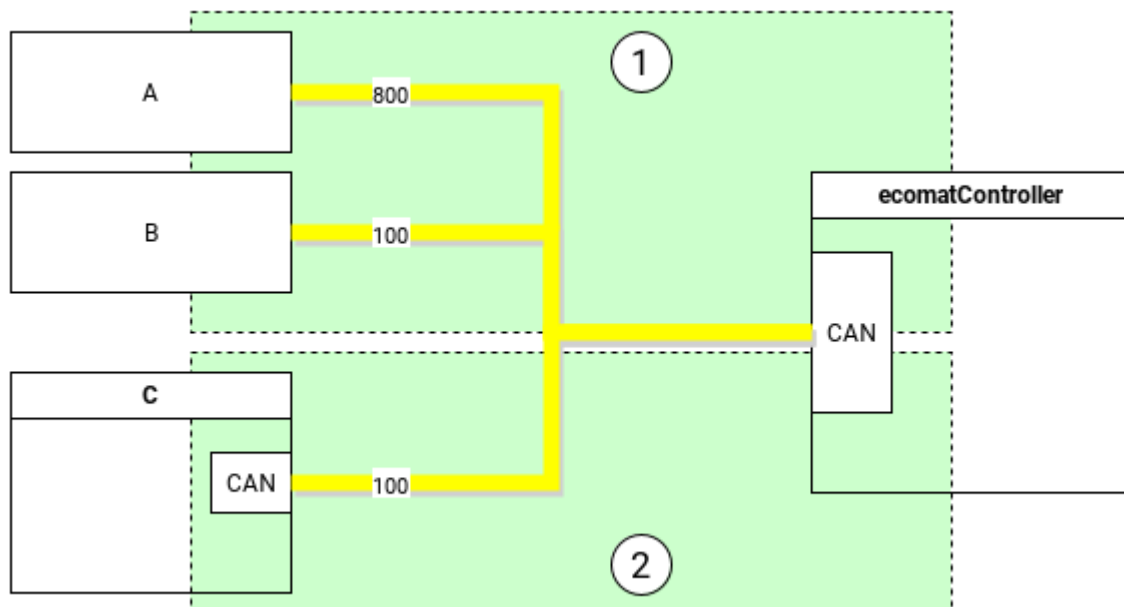
If more than 5000 safety-related messages (SRDOs) per second are transmitted for a safety function:

- ▶ Refer to the applicable standards for the calculation of the safety parameter PFH<sub>D</sub> for the safety function.

## Example

58745

Three CANopen safety devices (A, B, C) are connected to a CAN bus interface of the controller. The 3 CANopen safety devices belong to 2 safety functions (1 and 2), which are divided and configured as follows:



### 1: Safety function "Load moment limiter"

A: Encoder 1; 800 SRDOs / second

B: Encoder 2; 100 SRDOs / second

$A + B = 800 + 100 = 900 \text{ SRDOs / second} < 5000 \Rightarrow \text{OK}$

- ▶ In SISTEMA: Use a "CANopen Safety" subsystem with PFH<sub>D</sub> =  $1.0 \times 10^{-8}$  for the safety function "Load moment limiter".

### 2: Safety function "Emergency switching off"

C: E-stop; 100 SRDOs / second

$C = 100 \text{ SRDOs / second} < 5000 \Rightarrow \text{OK}$

- ▶ In SISTEMA: Use a "CANopen Safety" subsystem with PFH<sub>D</sub> =  $1.0 \times 10^{-8}$  for the safety function "Emergency switching off".



Determine the number of SRDOs / second:

- ▶ In the CODESYS project: Double click on the SIL2 device under: [CAN] > [CANbus] > [CANopen\_Manager\_SIL2]
- > The device properties appear.
- ▶ Click [SRDOs].
- ▶ Select SRDO.
- ▶ Click [Edit].
- > The [SRDO properties] dialogue appears.
- ▶ Under [Transmission settings ], determine the [Refresh Time (ms)].
  
- ▶ Calculation with one SRDO:  $1000 \text{ ms} / \text{Refresh Time (ms)} = \text{number of SRDOs / second}$
- ▶ Calculation with several SRDOs: Add the number of the individual SRDOs / second for each SRDO.

## 10 Set-up and maintenance

### Content

Connect the device to the network .....	180
Check the operating system version of the device .....	181
Update the operating system of the device .....	182
Transfer CODESYS project to device .....	184
Data transmission for series production .....	187
Documentation of the overall application .....	191
CODESYS Debugging.....	196

25063

### 10.1 Connect the device to the network

25071

To connect the device with the Ethernet network:

- ▶ Integrate the device according to the required topology via the Ethernet interfaces into the Ethernet network.  
 Description → **Ethernet interface** (→ p. [59](#))  
 Configuration → **Configure Ethernet interface** (→ p. [94](#))  
 Configure the programming interface → **Configure the programming interface** (→ p. [73](#))

25076



- ▶ Before downloading a software component, read out the serial number and consult the network diagram to make sure that you are accessing the right controller.



## 10.2 Check the operating system version of the device

### Content

Check the operating system version of the device.....	181
Check the hardware version of the device .....	181

39485

### 10.2.1 Check the operating system version of the device

39486

To check the operating system version of the device:

- ▶ Connect to PLC (→**Set communication path of PLC** (→ p. [73](#)))
- ▶ Copy the file \info\swinfo.txt by clicking on [<<] to a local PC drive (→**Manage files** (→ p. [81](#))).
- ▶ Open swinfo.txt in an editor, e.g. Notepad
- > Content of the opened file (example):
 

```
[ifmOS]
Version=V1.4.0.3
BuildDate=21.10.2016 15:11:14

[Bootloader]
Version=1.2.3.4
BuildDate=01.01.2017 23:10:15
```
- ▶ Check the information in the [ifmOS] area behind [Version=]
- > If the version deviates from the required version, the operating system must be updated.

### 10.2.2 Check the hardware version of the device

39484

To check the hardware version of the device:

- ▶ Connect to PLC (→**Set communication path of PLC** (→ p. [73](#)))
- ▶ Copy the file \info\devinfo.txt by clicking on [<<] to a local PC drive (→**Manage files** (→ p. [81](#))).
- ▶ Open and check devinfo.txt in an editor, e.g. notepad
- > devinfo.txt contains the following information: → **Hardware information (devinfo.txt)** (→ p. [193](#))

## 10.3 Update the operating system of the device

### Content

Update the operating system of the device with the ifm Maintenance Tool .....	182
Update the operating system of the device with the batch file .....	182

23501

### NOTICE!

Interruption of the voltage supply during the data transfer/update process

- > An incomplete operating system update may destroy the device.
- ▶ During the update process, the electric voltage supply of the device must be assured.
- ▶ Do not disconnect the device before the data transfer/update process is finished.

- ▶ Before the update process, please ensure:
  - Device is connected to the voltage supply and switched on
  - The Ethernet interface is connected to the same network as the PC

When loading the operating system to the controller, the controller verifies whether the software is compatible with the hardware. If they are not compatible, the transmission will stop.

### 10.3.1 Update the operating system of the device with the ifm Maintenance Tool

54480

To update the operating system of the device using the ifm Maintenance Tool:

→ Software manual "ecomatController AddIn for Maintenance Tool"

### 10.3.2 Update the operating system of the device with the batch file

23825

### NOTICE!

If the instructions on the screen are not complied with

- > The controller may be damaged.
- ▶ Strictly follow the instructions on the screen during the entire update process.

To update the operating system of the device with the batch file `update.bat`:

- ▶ Unpack the ZIP file with the batch file `update.bat` and the corresponding files in a local memory location.



Important: The data path in which the batch file is unpacked may not contain any blanks!

- ▶ Execute the batch file `update.bat`
- ▶ Follow the instructions on the screen.
- > The [ecomatController Update Start Menu] appears.

If the device is not set to the standard IP address 192.168.82.247:

- ▶ Use the [i] key to select the menu item [Set device ip-address].
- ▶ Enter the IP address set in the device and confirm with [RETURN].

- > The IP address setting in the batch program has been changed.
- ▶ Use the [P] key to call up the menu item [Ping device]
- > The ping command is executed. The device must answer to the ping request to enable an update process.

If ping is successful:

- ▶ Use the [0] key to call up the menu item [Continue update process].
- > Order number, software and hardware version are read from the device.

If an update is possible using the read data:

- > The [ecomatController Update Menu] appears.
- > Otherwise: Error message and return to the [ecomatController Update Start Menu].
- ▶ Follow the instructions on the screen.
- > The updated process is executed.

---

## **NOTICE!**

If the instructions on the screen are not complied with

- > The controller may be damaged.
- ▶ After loading the first file `cmd.ifm` and after the instruction on the screen, execute a power-on reset of the controller.

- ▶ After the power-on reset, continue the update process following the instructions on the screen.
- > The user is informed about the success of the update process.

If the update process is finished successfully:

- ▶ Disconnect the voltage supply.
- ▶ Re-connect the voltage supply after the waiting time.
- > The PLC boots with a new operating system.

## 10.4 Transfer CODESYS project to device

### Content

Load the standard application to the device .....	184
Load the safety application to the device .....	185
Delete the application program on the device .....	185

24872



- ▶ Familiarise yourself with the following CODESYS functions!
  - Translate the project/application and transfer them to the device  
→ Online help > [CODESYS Development System] > [Transferring Applications to the PLC]

25076



- ▶ Before downloading a software component, read out the serial number and consult the network diagram to make sure that you are accessing the right controller.

To save the CODESYS project on the device, transfer the following component:

- Standard application (→ **Load the standard application to the device** (→ p. [184](#)))
- Safety application (→ **Load the safety application to the device** (→ p. [185](#)))

When the application is loaded to the controller, the controller verifies whether the translated application is compatible with the installed firmware version and the configured memory layout. If they are not compatible, the transmission will stop.



- ▶ Observe notes on the operating modes of the Standard PLC of the device!  
→ **Overview of operating mode states** (→ p. [199](#))

### 10.4.1 Load the standard application to the device

24537



- When the transmission starts, the entire controller goes into the UPDATE mode. In this mode, all outputs of the device will be switched off and both applications (standard and safety application ) will be stopped.
- **Operating states** (→ p. [198](#))

To transfer the created application as boot project to the device:

#### Requirements:

- > Communication path is set (→ **Set communication path of PLC** (→ p. [73](#))).
- > Project tested.

#### 1 Translate application

- ▶ In the device tree: highlight application as active application.
- ▶ Translate the active application with [Build] > [Rebuild]
- > CODESYS generates the program code.

#### 2 Load application to the device

- ▶ Connect with the device via [Online] > [Login]
- > Active application is loaded to the device (download).
- > Application on the device is in the STOP state.

### 3 Start the application

- ▶ Start the application with [Debug] > [Start].
- > Application goes to the RUN state.

## 10.4.2 Load the safety application to the device

24538



### WARNING

If CODESYS connects itself with a controller, the controller goes into **non-safe operation**.

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function.
- ▶ Before accessing the controller, the project engineer must ensure that
  - the machine/plant does not constitute any danger.
  - the machine/plant is in a protected and closed environment.
  - no persons or objects are within the operational area of the machine/plant.



When the transmission starts, the entire controller goes into the UPDATE mode.

In this mode, all outputs of the device will be switched off and both applications (standard and safety application ) will be stopped.

→ **Operating states** (→ p. [198](#))

To transfer the created safety application as boot project to the device:

#### Requirements:

- > Communication path is set (→ **Set communication path of PLC** (→ p. [73](#))).
- > Project tested.

#### 1 Translate application

- ▶ In the device tree: highlight application as active application.
- ▶ Translate the active application with [Build] > [Rebuild]
- > CODESYS generates program code.

#### 2 Load application to the device

- ▶ Change to debug mode with [SIL2] > [Enter debug mode].
- ▶ Connect with the device via [Online] > [Login]
- > Active application is loaded to the device (download).
- > Application on the device is in the STOP state.

#### 3 Start the application

- ▶ Start the application with [Debug] > [Start].
- > Application goes to the RUN state.

## 10.4.3 Delete the application program on the device

39498

To delete an application stored on the device:

#### 1 Connect with the device

- ▶ In the device tree: highlight application as active application.
- ▶ Connect with the device with [Online] > [Login].
- > CODESYS is in the online mode.

## 2 Delete application

- ▶ In the editor window: Select tab [Device] > [Applications].
- ▶ Refresh view with [Refresh List].
- > List shows the applications that are stored on the device.
- ▶ Delete all applications in the device with [Remove All].  
OR:  
Highlight requested application and delete with [Remove] from the device.
- > The selected application will be deleted.

## 10.5 Data transmission for series production

### Content

Data transmission with the ifm Maintenance Tool.....	188
Transmission of the files with CODESYS.....	188
Data transmission with TFTP .....	189
Files for series production .....	189

23577

For the series production, application data and stored data can be transferred to the PC and then transferred from the PC to further devices.

The data transmission takes place in two steps:

1. Data backup from the device to the PC
2. Distribution of the backed up data to the target devices

When loading the operating system to the controller, the controller verifies whether the software is compatible with the hardware. If they are not compatible, the transmission will stop.

When loading the application to the controller, the compatibility of the application will be checked when it is loaded. Starting an incompatible application is not possible.

When other files are loaded, the compatibility will not be checked.



### WARNING

Incompatibility between files and hardware / software.

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function.
- ▶ The system manufacturer must ensure the compatibility of the following data:
  - iomapping.cfg with the device and with the IEC application.
  - memconf.cfg with the device and with the IEC application.
  - comconf.cfg with the device and with the IEC application.
  - Retain files and memory data to the IEC application. Recommendation: Use the checksum CRC32 in the file swinfo.txt for this purpose.
  - User files (that are read and written from the IEC application) with the IEC application.

25444



### WARNING

If the controller is restarted and especially in case of a power-on reset from the debug mode:

- > Risk of personal injuries and/or damage to property.
- > The controller passes into the operating mode.
- > The controller application starts.
- ▶ Before restarting the controller, the project engineer must ensure
  - that the machine/plant does not constitute any danger
  - that the machine/plant is in a protected and closed environment
  - that no persons or objects are within the operational area of the machine/plant
- ▶ Implement restart protection, start acknowledgement in the application.

## 10.5.1 Data transmission with the ifm Maintenance Tool

54488



When the transmission starts, the entire controller goes into the UPDATE mode.

In this mode, all outputs of the device will be switched off and both applications (standard and safety application ) will be stopped.

→ **Operating states** (→ p. [198](#))

Procedure to transmit data using the Maintenance Tool: → Software manual "ecomatController AddIn for Maintenance Tool"

## 10.5.2 Transmission of the files with CODESYS

24829  
24828

When the transmission starts, the entire controller goes into the UPDATE mode.

In this mode, all outputs of the device will be switched off and both applications (standard and safety application ) will be stopped.

→ **Operating states** (→ p. [198](#))




For this action, you need to sign in as device user.

For more user convenience:

- ▶ Create a user in the project and sign the user in once
  - → **Create a user in the CODESYS project** (→ p. [77](#))
  - → **Sign the user in to the CODESYS project** (→ p. [78](#))

To transfer files between the PC and the device:

### 1 Select file view

- ▶ In the device tree: Double-click on the symbol [CR7xxS (CR7xxS)]
- ▶ In the editor window: Select the [Files] tab.
- ▶ Click on the symbol  [Refresh]
- > The editor window shows the folder structure on the PC on the left and on the device on the right

### 2 Transfer file from PC to device

- ▶ Highlight the file on the left
- ▶ Select device target directory on the right
- ▶ Start transfer using the [>>] button
- > The file is transferred to the device

### 3 Transfer the file from the device to the PC

- ▶ Highlight the file on the right
- ▶ Select PC target directory on the left
- ▶ Start transfer using the [<<] button
- > The file is transferred to the PC



### 10.5.3 Data transmission with TFTP

23580



When the transmission starts, the entire controller goes into the UPDATE mode.

In this mode, all outputs of the device will be switched off and both applications (standard and safety application ) will be stopped.

→ **Operating states** (→ p. [198](#))

With the aid of the program TFTP, files can be transferred.

#### Transfer file from device to PC:

```
tftp -i IP address GET source target
```

IP address = address of the source device, e.g. 192.168.82.247

Source = source file on the device

Target = target file on the PC

#### Transfer file from PC to device:

```
tftp -i IP address PUT source target
```

IP address = address of the source device, e.g. 192.168.82.247

Source = source file on the PC

Target = target file on the device

#### Example:

```
tftp -i 192.168.82.247 PUT [Windows-Pfad]\ifmOS.ifm /os/ifmOS.ifm
```

### 10.5.4 Files for series production

39508

The following files must be transferred:

Data name / path	Description
<b>apps</b>	Folder
▪ standard.app	Application non-safe
▪ safe.app	Application safe
<b>os</b>	Folder
▪ ifmOS.ifm	ifmOS
<b>cfg</b>	Folder
▪ comconf.cfg	Communication configuration
▪ memconf.ifm	Memory configuration

The following file must be transferred according to the kind of application (retain data and free user data):

Data name / path	Description
<b>retain</b>	Folder

▪ standard.ret	Application retain non-safe
▪ standard.mb	Application memory bytes non-safe
▪ safe.ret	Application retain safe
▪ safe.mb	Application memory bytes safe
<b>data</b>	Folder
▪ *.*	Memory space for user-defined data

## 10.6 Documentation of the overall application

### Content

Document the serial number .....	191
Create a network diagram .....	191
Document the checksums .....	191
Read the device information .....	192
Display system information .....	195

25035

The project engineer must read the parameters of the safety application back, document them and compare them with the expected settings (validation). This is the only way to ensure that the safety application corresponds with the expected settings.

### 10.6.1 Document the serial number

25077

All safety controllers have a unique serial number.

- ▶ Document and archive this serial number together with the machine ID and the location where the machine is operated before installing the device!

Only then will it be possible to revise specific machines in case of errors at a later point in time and to replace components.

### 10.6.2 Create a network diagram

25075

- ▶ In the user's production facility, draw a diagram of the controller network in the machine.
  - ▶ Enter the network of each installed controller in the network diagram.
  - ▶ Enter the serial number of each installed controller into the network diagram.

### 10.6.3 Document the checksums

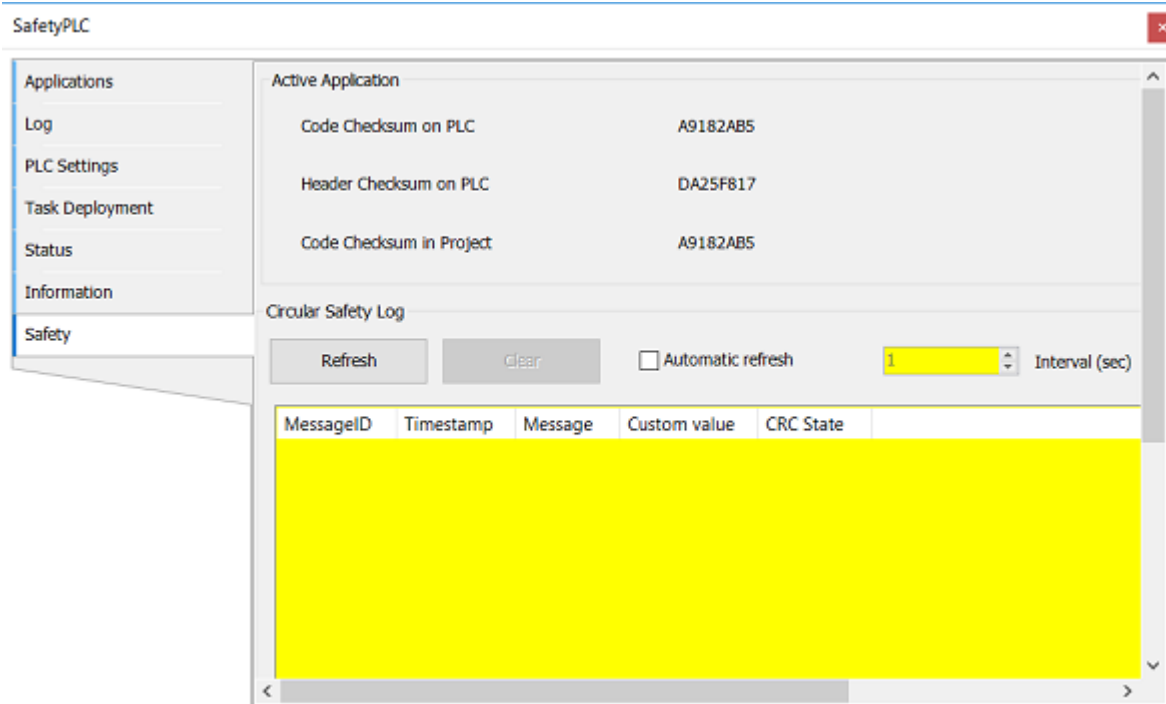
24750

While the safety application is translated, CODESYS generates a code checksum (CodeCRC) from the application code.

In addition, CODESYS generates a header checksum (HeaderCRC) on the PLC from the data of the application header.

This ensures that changes on the safety application can be detected because of changes in the checksums.

CODESYS shows the checksums in the device editor of the safety PLC in the tab [Safety] under [Active Application]:



► Please document the checksums when the safety application is being certified!

## 10.6.4 Read the device information

25177

Information of the device are provided in different files on the controller in the directory \info. This information can be used to identify the device and to determine its version and software content.

The following files with information are included:

file name	description
appinfo.txt	→ <b>Application information (appinfo.txt)</b> (→ p. <a href="#">193</a> )
devinfo.txt	→ <b>Hardware information (devinfo.txt)</b> (→ p. <a href="#">193</a> )
swinfo.txt	→ <b>Software information (swinfo.txt)</b> (→ p. <a href="#">194</a> )

- To check/document the information in the device:
  - Connect to PLC (→ **Set communication path of PLC** (→ p. [73](#)))
  - Copy the files that are listed in the table in the directory \info by clicking on [<<] to a local PC drive (→ **Manage files** (→ p. [81](#))).
  - Open the files in a text editor and check their content



Further information about files and directory structure on the controller:  
→ **Directory structure and file overview** (→ p. [485](#))

## Application information (appinfo.txt)

25037

The file appinfo.txt includes the following information:

Section	Key	Description
Standard task X	Task type	Standard application: Task type
Standard task X	Task priority	Standard application: Task priority
Standard task X	Task interval	Standard application: Task interval in ms (only if type = cyclic)
Standard task X	WatchdogEnable	Standard application: Watchdog enabled (TRUE, FALSE)
Standard task X	WatchdogTime	Standard application: Watchdog time in ms
Standard task X	WatchdogSensitivity	Standard application: Watchdog sensitivity
Safe-Task-X	Task type	Safety application: Task type
Safe-Task-X	Task priority	Safety application: Task priority
Safe-Task-X	Task interval	Safety application: Task interval in ms (only if type = cyclic)
Safe-Task-X	WatchdogEnable	Safety application: Watchdog enabled (TRUE, FALSE)
Safe-Task-X	WatchdogTime	Safety application: Watchdog time in ms
Safe-Task-X	WatchdogSensitivity	Safety application: Watchdog sensitivity
Security	PasswordsActive	Passwords are set for all application users (TRUE, FALSE)

with X = 0...3, depending on the number of the CODESYS task



► Please document the content of the file appinfo.txt when the safety application is being certified!



Read the files from the controller:

→ **Read the device information** (→ p. [192](#))

## Hardware information (devinfo.txt)

25180

The file devinfo.txt includes the following information:

Section	Key	Description
IDENTIFICATION	Serial	Serial number of the device
DEVICE	Order no.	ifm article number of the device
DEVICE	Name	ifm article description of the device
DEVICE	Revision	Production status of the device
MANUFACTURE	Date	Production data and time
HARDWARE	Version	Hardware version of the device
NETWORK	MAC	MAC address of the device
NETWORK	DNS	DNS name
CERTIFICATE	Certificate [0..9]	Certificates of the controller
CRC	CRC32	CRC32 over the entire previous ASCII range. IEEE 802.3 Standard Ethernet Polynom. This section must be at the end of the file.



► Please document the content of the file `devinfo.txt` when the safety application is being certified!



Read the files from the controller:  
→ **Read the device information** (→ p. [192](#))

## Software information (swinfo.txt)

25181

The file `swinfo.txt` contains an individual section with current values for each of the following software components. If the corresponding software component is not included in the controller, there is no corresponding section in the file.

Software components:

- ifmOS
- Bootloader
- SIS-SYS
- comconf
- memconf
- iomapping
- StandardPLC
- SafePLC
- StandardRetain
- StandardMB
- SafeRetain
- SafeMB

The file `swinfo.txt` contains the following information for the above-mentioned software components:

Section	Key	Description
Software	Name	Software name (e.g. ifmOS)
Software	Title	only for standard PLC and safety PLC: Information from CODESYS project title
Software	Description	only for standard PLC and safety PLC: Information from Application Information Description (for both applications, separate information is possible)
Software	Version	Software version number (e.g. V1.5.3.29)
Software	Build date	Software build date (format DD.MM.YYYY, hh:mm:ss)
Software	CodeCRC32	CRC32 via the software. Standard Ethernet Polynom (04C11DB71H).
Software	HeadCRC32	CRC32 via the header of the software of the Standard Ethernet Polynom software (04C11DB71H).
Software	Active	only for boot loader and ifmOS: indicates which software is executed (TRUE/FALSE).
CRC	CRC32	CRC32 over the entire previous ASCII range up to section [CRC] (last character is ]). IEEE 802.3 Standard Ethernet Polynom (04C11DB71H). This section must be at the end of the file.



- ▶ The file indicates the current value (versions, names, CRCs, etc.) of the values in the controller.
- ▶ Please document the content of the file `swinfo.txt` when the safety application is being certified!



Read the files from the controller:

→ **Read the device information** (→ p. [192](#))

## 10.6.5 Display system information

39514

In the online mode the device tree displays the current values of the following system parameters:

Parameter	Description	Possible values
[IP Settings]	IP settings	--
▪ [IP Address]	IP address of the device	E.g. 192.168.0.100
▪ [IP Mask]	Subnet mask of the network	E.g. 255.255.255.0
▪ [Gateway Address]	IP address of the network gateway	E.g. 192.168.0.2
[Version Firmware]	Version of the installed firmware	E.g. V1.4.0
[Serial Number Device]	Serial number of the device	E.g. 1511AB019

To display the system information of the device:

- ▶ Establish connection between CODESYS and CR7xxS.
- ▶ Select [Online] > [Login].
- > CODESYS changes to the online mode.
- ▶ In the device tree: Double-click on [System\_Info]
- ▶ In the editor window: Select tab [Parameter].
- > In the editor window: Table shows current values of the system parameters.

## 10.7 CODESYS Debugging

25079



- ▶ Familiarise yourself with the following CODESYS functions!
  - CODESYS Debugging
    - Online help > CODESYS Development System > Test and remove errors



### WARNING

If CODESYS connects itself with a controller, the controller goes into **non-safe operation**.

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function.
- ▶ Before accessing the controller, the project engineer must ensure that
  - the machine/plant does not constitute any danger.
  - the machine/plant is in a protected and closed environment.
  - no persons or objects are within the operational area of the machine/plant.

25444



### WARNING

If the controller is restarted and especially in case of a power-on reset from the debug mode:

- > Risk of personal injuries and/or damage to property.
- > The controller passes into the operating mode.
- > The controller application starts.
- ▶ Before restarting the controller, the project engineer must ensure
  - that the machine/plant does not constitute any danger
  - that the machine/plant is in a protected and closed environment
  - that no persons or objects are within the operational area of the machine/plant
- ▶ Implement restart protection, start acknowledgement in the application.

When debugging is enabled:

- > CODESYS resets are possible ( → **Reset** (→ p. [204](#))):
  - > Reset (warm)
  - > Reset (cold)
  - > Reset (default)
- > When executing a reset:
  - > all resources opened by the IEC application in question (files, sockets, etc.) will be closed.
  - > all inputs and outputs assigned to the IEC application in question will be switched off and put into default mode.
- > It is possible to force values in the IO mapping.
- > It is possible to start and stop the IEC application.
- > Executing the following debugging functions:
  - > Use of up to 8 breakpoints per PLC
  - > Running a program step by step (menu [Debug] > [Step Over], [Step Into], [Step Out], [Run to Cursor])
  - > for IEC application exception: jump to the position in the editor





#### CODESYS debugging restrictions

- ▶ Use a maximum of 6 breakpoints for step-by-step program execution (2 breakpoints are used internally by CODESYS).
- When the error message [Breakpoint could not be set on target.] appears in CODESYS:
  - ▶ Disable breakpoints or log out and log in again to release breakpoints.
- In step-by-step program execution, CODESYS uses 1 breakpoint internally for each CASE branch in CASE instructions:
  - ▶ Debug CASE instructions with more than 8 branches only as follows:
    - Set breakpoint at the CASE statement.
    - Determine the target branch using the variable value.
    - Set breakpoint in target branch.
    - Press [Debug] > [Start] to continue debugging.

# 11 Operation

## Content

Operating states .....	198
Status LEDs.....	201
Reset .....	204

25069

## 11.1 Operating states

### Content

Overview of operating mode states .....	199
---	-----

60525

The following table describes the possible operating modes of the device:

Operating status	Description	Has an impact on...	condition of the inputs	condition of the outputs
POWER_OFF	The device is switched off. Initial condition.	Whole device	OFF	OFF
INIT	Starting the device: Initialising the software, checking the hardware	Whole device	OFF	OFF
RUNTIME_OPERATING	Normal operating status of the controller Executing the application.	Standard or safety PLC	ACTIVE	ACTIVE
RUNTIME_STOP	Application is stopped after an error class B event (SERIOUS ERROR).	Standard or safety PLC	INACTIVE	OFF
RUNTIME_DEBUG_RUN	Executing the application in the debug mode.	Standard or safety PLC	ACTIVE	OFF
RUNTIME_DEBUG_BP_HALT	The application is in the debug mode, stop through breakpoint or exception.	Standard or safety PLC	INACTIVE	OFF
RUNTIME_DEBUG_STOP	The application is stopped in the debug mode.	Standard or safety PLC	INACTIVE	OFF
SHUTDOWN	Shutting down the device: Closing the applications, storing non-volatile data and checking the hardware.	Whole device	OFF	OFF
UPDATE	Updating the device: loading data to the device.	Whole device	OFF	OFF
SYSTEM_STOP	The device is stopped after an error class A event A = (FATAL ERROR).	Whole device	OFF	OFF
SLEEP	Sleep Mode The sleep mode is supported by devices with hardware version >= V1.0.5.2.	Whole device	OFF	OFF

Legend:

OFF: Inputs are in the default mode and outputs are switched off.

ACTIVE: The assigned inputs and outputs are operated with the configuration set by the IEC application (mode, filter, diagnostics, etc.).

INACTIVE: The assigned inputs include their preset configuration and will continue to be read. The diagnostic in the IEC application is suppressed. The hardware is still protected.

### 11.1.1 Overview of operating mode states

24539

The following figure shows the possible operating modes of the device.

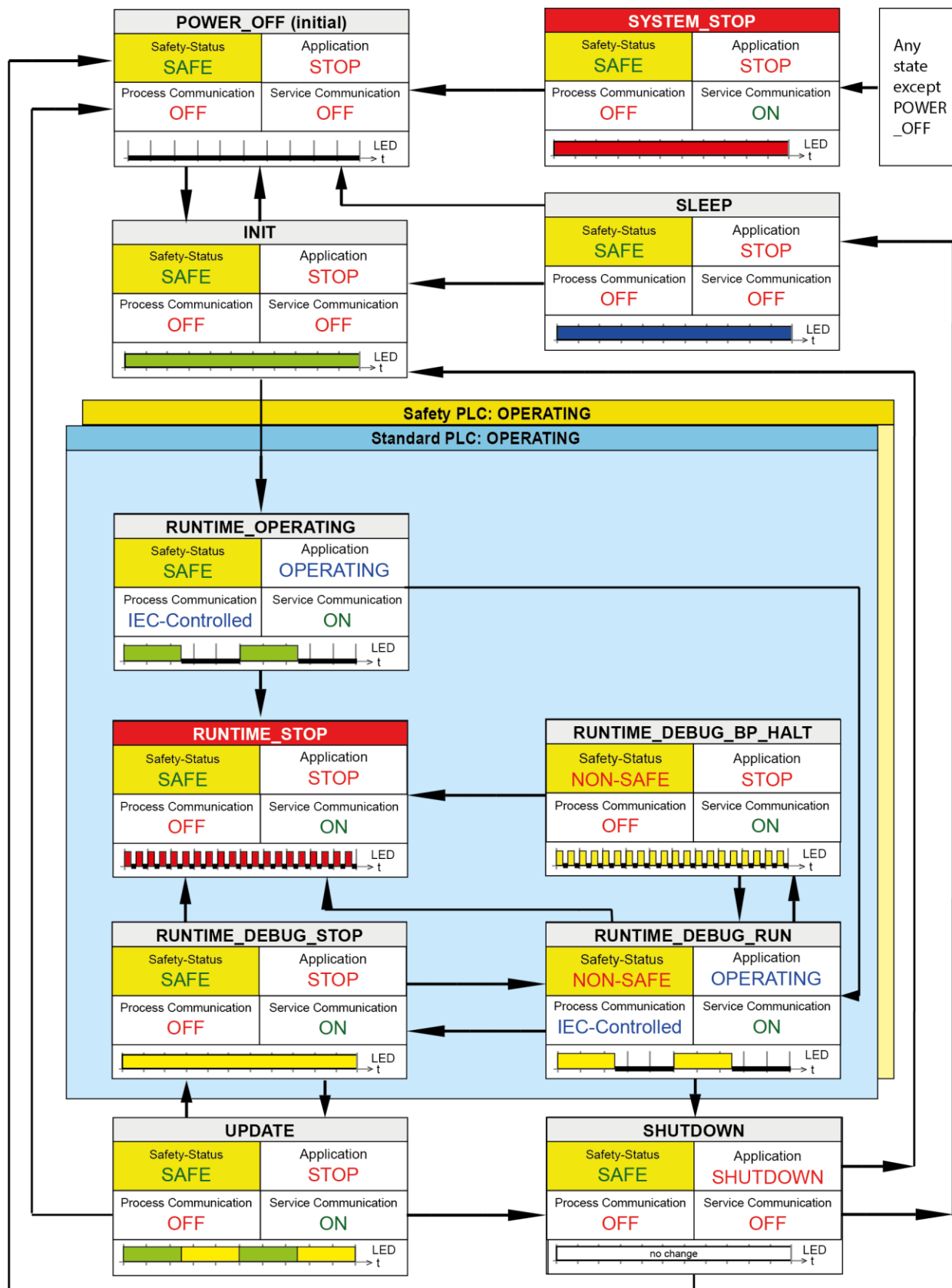
It contains:

- safety status of the controller
- status of the application
- status of the process communication (inputs/outputs, CAN bus)
- status of the service communication (connection with the programming device)
- display of the LEDs SYS0 / SYS1

All states within

- standard PLC OPERATING
- safety PLC OPERATING

are independent and free of interaction to the states in the corresponding PLC that is changed. Any states outside of this, concern both PLCs at once.



## 11.2 Status LEDs

### Content

Status LED: system ifm operating system (SYS0+SYS1).....	201
Status LED: system PLC (SYS0, SYS1).....	201
Status LED: System bootloader (SYS0).....	202
Status LED: Sleep mode (SYS0) .....	203
Status LED: Ethernet interfaces (ETH0, ETH1) .....	203
Controlling LEDs in the applications.....	203

39586





The device has the following LEDs:

LED	Description
SYS0	Status of the standard PLC Status of the ifm operating system Status of the bootloader
SYS1	Status of the safety PLC Status of the ifm operating system
ETH0	Status of the Ethernet interface 0
ETH1	Status of the Ethernet interface 1
APPL0 APPL1 APPL2 APPL3	LEDs for free use in the application

### 11.2.1 Status LED: system ifm operating system (SYS0+SYS1)

39581

For the status of the ifm operating system, both LEDs SYS0 and SYS1 are lit simultaneously:

LED colour	Display	Description
off	permanently off 	ifm operating system on the unit: POWER_OFF
Green	permanently on 	ifm operating system on the unit: INIT
Red	permanently on 	ifm operating system on the unit: SYSTEM_STOP Error class = A
no change	no change no change	ifm operating system on the unit: SHUTDOWN
green-yellow	Flashing with 2 Hz  (time frame = 200 ms)	ifm operating system on the unit: UPDATE

### 11.2.2 Status LED: system PLC (SYS0, SYS1)

39582

The SYS0 LED is for the "standard PLC".

The SYS1 LED is for the "safety PLC".

The status of one of the PLCs has no influence on the display of the other PLC.

LED colour	Display	Description
Green	permanently on	RUNTIME_OPERATING no application loaded
		
Green	Flashing with 2 Hz	RUNTIME_OPERATING Application = RUN
	 (time frame = 200 ms)	
Yellow	Flashing with 2 Hz	RUNTIME_DEBUG_RUN Application = RUN
	 (time frame = 200 ms)	
Yellow	Flashing with 10 Hz	RUNTIME_DEBUG_BP_HALT Application = STOP
	 → t	
Yellow	permanently on	RUNTIME_DEBUG_STOP Application = STOP
		
Red	flashes with 10 Hz	RUNTIME_STOP Error class = B
	 → t (time frame = 200 ms)	

### 11.2.3 Status LED: System bootloader (SYS0)



39580

The SYS0 LED is for the bootloader status only.  
The SYS1 LED is switched off in these cases.

39580



Only execute the bootloader update when explicitly requested by ifm!


LED colour	Display	Description
Green	Flashing with 5 Hz	no runtime system loaded
	 (time frame = 200 ms)	
green-yellow	flashes with 5 Hz	Bootloader update process active
	 → t (time frame = 200 ms)	

## 11.2.4 Status LED: Sleep mode (SYS0)

60526

Only LED SYS0 is used for the sleep mode status.  
LED SYS1 is switched off in these cases.



The sleep mode is supported by devices with hardware version  $\geq$  V1.0.5.2.

LED colour	Display	Description
Blue	Permanently on	Sleep mode is active
		

## 11.2.5 Status LED: Ethernet interfaces (ETH0, ETH1)

39585

The two Ethernet interfaces indicate their status as follows:

LED colour	Display	Description
Green	permanently on	Ethernet connection is established non data traffic
		
Green	blinks	Ethernet connection is established with data traffic
		

## 11.2.6 Controlling LEDs in the applications

39477

The LEDs APPL0 to APPL3 are for free use in the applications.  
This is the function of the FB **SetLED** (→ p. [276](#)).

Possible colours: → **LED\_COLOUR (ENUM)** (→ p. [213](#))

Possible frequencies: → **LED\_FLASH\_FREQ (ENUM)** (→ p. [213](#))

## 11.3 Reset

### Content

Reset behaviour of the system.....	204
Executing reset variants .....	205
Reset application (warm).....	205
Reset application (cold).....	205
Reset application (origin).....	205

39674

### 11.3.1 Reset behaviour of the system

60529

The system shows the following behaviour for the different reset variants and system resources:

Resource State	CODESYS reset (warm) (only the PLC concerned)	CODESYS reset (cold) (only the PLC concerned)	CODESYS reset (origin) (only the PLC concerned)	PowerOn Reset (standard and safety application)	Factory reset (standard and safety application)
IEC application	is kept	is kept	is deleted	is kept	is deleted
IEC Variables	are re-initialised	are re-initialised	are deleted	are re-initialised	are deleted
IEC-persistent (non-volatile variables)	are kept	are kept	are deleted	are kept	are deleted
IEC memory bytes	are kept	are kept	are kept	are kept	are deleted
user files	are kept	are kept	are kept	are kept	are deleted
iomapping.cfg	is kept	is kept	is kept	is kept	is deleted
comconf.cfg	is kept	is kept	is kept	is kept	is deleted (The comconf.cfg is restored with default values after deletion)
memconf.ifm	is kept	is kept	is kept	is kept	is deleted
passwd.ifm	is kept	is kept	is kept	is kept	is deleted
Condition after reset	RUNTIME_ DEBUG_STOP	RUNTIME_ DEBUG_STOP	RUNTIME_ DEBUG_STOP	RUNTIME_ OPERATING	RUNTIME_ OPERATING No application



Definition "are re-initialised": The variable will be set to its initialisation value.

If the programmer does not assign an initialisation value to one of the variables, CODESYS will initialise this variable with the standard value (usually with "0").



### 11.3.2 Executing reset variants

60530

The following reset variants can be called up as follows:

Calling component	CODESYS reset (warm)	CODESYS reset (cold)	CODESYS reset (origin) (Only the PLC concerned goes into the Update state)	PowerOn Reset (standard and safety application)	Factory reset (standard and safety application)
CODESYS in Debug Mode	yes	yes	yes	no	no
IEC application with FUN reset	no	no	no	yes	no
Maintenance Tool / Update Tool	yes	yes	yes	yes	yes

yes = reset call possible  
no = reset call not possible

### 11.3.3 Reset application (warm)

39671

To reset the application:

- ▶ In the device tree: Select [Application] and select
- ▶ [Online] > [Login] as active application.
- > CODESYS changes to the online mode.
- ▶ Select [Online] > [Reset warm] to reset the application.
- > Application changes to the STOP state.
- > Standard variables are newly initialised.
- > Retain variables keep their values.

### 11.3.4 Reset application (cold)

39675

To reset the application:

- ▶ In the device tree: Select [Application].
- ▶ Select [Online] > [Login].
- > CODESYS changes to the online mode.
- ▶ Select [Online] > [Reset cold] to reset the application.
- > Application changes to the STOP state.
- > All variables are newly initialised

### 11.3.5 Reset application (origin)

25145

To reset the application:

- ▶ In the device tree: Select [Application].
- ▶ Select [Online] > [Login].
- > CODESYS changes to the online mode.
- ▶ Select [Online] > [Reset origin] to reset the application.
- > Application changes to the STOP state and is deleted.
- > All variables are newly initialised

- > Non-volatile variables will be set to 0.
- > PLC is reset to the original state.

## 12 ifm function libraries

### Content

General information .....	207
Using the function blocks.....	207
Device library .....	211
CAN libraries .....	216
Input and output libraries .....	252
Help function libraries .....	317
Safety libraries .....	321

27260

This chapter contains the detailed description of the function libraries provided by ifm electronic for programming the device under CODESYS 3.5.

### 12.1 General information

27067

General information about:

- → **Messages / diagnostic codes of the function blocks** (→ p. [463](#))
- → **ifm behaviour models for function blocks** (→ p. [497](#))

### 12.2 Using the function blocks

25498

The following table shows in which PLC and in which PRG a function block from an ifm library may be used:



In contrast to the PLCopen Safety specification, safe function blocks can also be used in the standard PLC. However, no safety functions can be executed on the standard PLC via these function blocks.

- Only execute the safety function on the safety PLC.

Library	POU / function	Standard PLC Standard PRG	Safety PLC Standard PRG	Safety PLC Safety PRG
ifmDevice	Reset	X	X	---
ifmRawCAN	CAN_Enable	X	X	---
	CAN_Recover	X	X	---
	CAN_RemoteRequest	X	X	---
	CAN_RemoteResponse	X	X	---
	CAN_Rx	X	X	---
	CAN_RxMask	X	X	---
	CAN_RxRange	X	X	---
	CAN_RxRangeExt	X	X	---
	CAN_Status	X	X	---

Library	POU / function	Standard PLC Standard PRG	Safety PLC Standard PRG	Safety PLC Safety PRG
	CAN_Tx	X	X	---
ifmCANopenManager	COP_GetNodeState	X	X	---
	COP_SDORead	X	X	---
	COP_SDOWrite	X	X	---
	COP_SENDRMT	X	X	---
ifmConfigSwThreshold	ConfigSwThreshold	X	X	X
ifmFastInput	FastCount	X	X	X
	IncEncoder	X	X	X
	Period	X	X	X
ifmIOcommon	Input	X	X	X
	Output	X	X	X
	SetLED	X	X	X
	SupplySwitch	X	X	X
	SystemSupply	X	X	X
	Temperature	X	X	X
ifmIOconfigDiagProt	ConfigDiagLevel	X	X	X
	ConfigDiagProt	X	X	X
ifmOutGroup	OutputGroup	X	X	X
ifmOutHBridge	HBridge	X	X	X
ifmOutPWM	CurrentControl	X	X	X
	PWM1000	X	X	X
ifmSysInfo	GetInfo	X	X	---
ifmIOSafety	SF_Input	X	X	X
	SF_InputBlanking	X	X	X
	SF_Output	X	X	X
	SF_OutGroup	X	X	X
	SF_CurrentControl	X	X	X
	SF_PWM1000	X	X	X

Library	POU / function	Standard PLC Standard PRG	Safety PLC Standard PRG	Safety PLC Safety PRG
	SF_HBridge	X	X	X
	SF_OutputEnh	X	X	X
	SF_OutGroupEnh	X	X	X
	SF_CurrentControlEnh	X	X	X
	SF_PWM1000Enh	X	X	X
	SF_HBridgeEnh	X	X	X
ifmPLCopenAddonSafe	SF_Equivalent_BOOL	X	X	X
	SF_Antivalent_BOOL	X	X	X
	SF_Equivalent_DINT	X	X	X
	SF_Equivalent_UINT	X	X	X
	SF_Equivalent_UDINT	X	X	X
	SF_Equivalent_REAL	X	X	X
ifmPLCopenSafe	SF_Antivalent	X	X	X
	SF_CamshaftMonitor	X	X	X
	SF_DoubleValveMonitoring	X	X	X
	SF_EDM	X	X	X
	SF_EmergencyStop	X	X	X
	SF_EnableSwitch	X	X	X
	SF_Equivalent	X	X	X
	SF_ESPE	X	X	X
	SF_FootSwitch	X	X	X
	SF_GuardLocking	X	X	X
	SF_GuardMonitoring	X	X	X
	SF_ModeSelector	X	X	X
	SF_OutControl	X	X	X
	SF_SafetyRequest	X	X	X
	SF_SingleValveCycleMonitoring	X	X	X
	SF_SingleValveMonitoring	X	X	X

Library	POU / function	Standard PLC Standard PRG	Safety PLC Standard PRG	Safety PLC Safety PRG
	SF_TwoHandControlTypeII	X	X	X
	SF_TwoHandControlTypeIII	X	X	X
	SF_ValveGroupControl	X	X	X

Legend:

X = may be used

--- = may not be used

## 12.3 Device library

### Content

Library ifmDeviceCR07nn.library .....	211
---------------------------------------	-----

27256

### 12.3.1 Library ifmDeviceCR07nn.library

#### Content

CAN_BAUDRATE (ENUM) .....	212
CAN_CHANNEL (ENUM) .....	212
CANconstants (GVL) .....	212
SysInfo (GVL) .....	212
SysInfoStruct (STRUCT) .....	213
LED_COLOUR (ENUM) .....	213
LED_FLASH_FREQ (ENUM) .....	213
Reset .....	214
RESET_TYPE (ENUM) .....	215

27152

The library contains the following:

- device-specific data structures
- device-specific enumeration types
- device-specific global variables and constants
- device-specific functions

**CAN\_BAUDRATE (ENUM)**

27157

Name	Description	Possible values		Data type	Value
CAN_BAUDRATE	Data transmission rate of the CAN interface	KBAUD_20	20 kilobaud	INT	20
		KBAUD_33	33.3 kilobaud	INT	33
		KBAUD_50	50 kilobaud	INT	50
		KBAUD_83	83.3 kilobaud	INT	83
		KBAUD_100	100 kilobaud	INT	100
		KBAUD_125	125 kilobaud	INT	125
		KBAUD_250	250 kilobaud	INT	250
		KBAUD_500	500 kilobaud	INT	500
		KBAUD_666	666.6 kilobaud	INT	666
		KBAUD_800	800 kilobaud	INT	800
		KBAUD_1000	1000 kilobaud	INT	1000

**CAN\_CHANNEL (ENUM)**

27159

Name	Description	Possible values		Data type	Value
CAN_CHANNEL	Identifier of the CAN Interface	CHAN_0	CAN interface 0	INT	0
		CHAN_1	CAN interface 1	INT	1
		CHAN_2	CAN interface 2	INT	2
		CHAN_3	CAN interface 3	INT	3

**CANconstants (GVL)**

27170

Name	Description	Data type	Value
usiNumberCANItf	Number of the CAN interfaces of the devices	UINT	4

**SysInfo (GVL)**

27351

Name	Description	Data type	Value
usiNumberOfSysInfo	Number of system components of the device	USINT	8
aSysInfoList	Variable with list of the system components (→ <b>aSysInfoList (GVL)</b> (→ p. <a href="#">320</a> ))	ARRAY[0..8] OF <b>SysInfoStruct (STRUCT)</b> (→ p. <a href="#">213</a> )	



**SysInfoStruct (STRUCT)**

27352

Designation	Data type	Description	Possible values
eInfoType	INFO_TYPE	System component	E.g. FIRMWARE_DEVICE
sValue	STRING (255)	Value of the system component	E.g. 3.1
sName	STRING (32)	Name of the system component	E.g. FW Device

**LED\_COLOUR (ENUM)**

27280

Name	Description	Possible values		Data type	Value
LED_COLOUR	Colour of the LED (RGB code)	BLACK (OFF)	Off	UINT	0x00 0000
		WHITE	White	UINT	0xFF FFFF
		RED	Red	UINT	0xFF 0000
		GREEN	Green	UINT	0x00 FF00
		BLUE	Blue	UINT	0x00 00FF
		YELLOW	Yellow	UINT	0xFF FF00
		MAGENTA	Magenta	UINT	0xFF 00FF
		CYAN	Cyan	UINT	0x00 FFFF

**LED\_FLASH\_FREQ (ENUM)**

27281

Name	Description	Possible values		Data type	Value
LED_FLASH_FREQ	Flashing frequency of the status LED	FRQ_0Hz	off	INT	0
		FRQ_05Hz	0.5 Hz	INT	1
		FRQ_1Hz	1 Hz	INT	2
		FRQ_2Hz	2Hz	INT	4
		FRQ_5Hz	5 Hz	INT	7
		FRQ_10Hz	10Hz	INT	8

## Reset

54512

**Function block type:** Function (FUN)  
**Library:** ifmDeviceCR07nn.library  
**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

54513

The FUN reset is used to set or execute different types of PLC and system resets.

## Input parameter

54514

Parameter	Data type	Meaning	Possible values
eResetTypes	RESET_TYPE	Setting the reset type	→ <b>RESET_TYPE (ENUM)</b>

## Output parameter

54515

Parameter	Data type	Meaning	Possible values
Reset	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes:)

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_DONE                      State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_UNDEFINED                Error: Unknown error  
     ► Contact the ifm Service Center!
- ERR\_NOT\_SUPPORTED          Error: Invalid function calls; Function is not supported.
- ERR\_ACCESS                    Error: FB/Function cannot access the required resource; Resource is blocked by another task.
- DIAG\_ACCESS                  FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.

**RESET\_TYPE (ENUM)**

60531

Name	Description	Possible values		Value
RESET_TYPE	Reset mode	NONE	No action. Default value	0
		POWER	Power-on reset will be immediately executed.	1
		POWER_RECOVER	Restarting the device after it has been switched off (VBB < 5.5V), if VBB reaches more than 8V again.	2
		WARM *	Warm start (PLC reset) will be executed immediately.	3
		COLD *	Cold start (PLC reset) will be executed immediately.	4
		ORIGIN *	Reset to original state (PLC reset) will be executed immediately.	5
		FACTORY *	The factory reset for standard and safety PLC will be immediately executed.	6
		SLEEP	The sleep mode is supported by devices with hardware version >= V1.0.5.2. The unit switches to sleep mode when the function is executed. After switching VBB15 off and on again, the control unit will reboot (PowerOn Reset).	7

The reset modes marked with a star \* are currently not supported.

## 12.4 CAN libraries

### Content

Library ifmRawCAN.library .....	216
Library ifmCANopenManager.library .....	243

27169

### 12.4.1 Library ifmRawCAN.library

#### Content

CAN_Enable .....	217
CAN_Recover .....	219
CAN_RemoteRequest .....	221
CAN_RemoteResponse .....	223
CAN_Rx .....	225
CAN_RxMask .....	228
CAN_RxRange .....	231
CAN_RxRangeExt .....	234
CAN_Status .....	237
CAN_Tx .....	240
CAN_Info (GVL) .....	242
CAN_BUS_STATE (STRUCT) .....	242

27288

The library contains POUs and data structures for the programming of the CAN Layer 2 level of the CAN interfaces of the device under CODESYS.

## CAN\_Enable

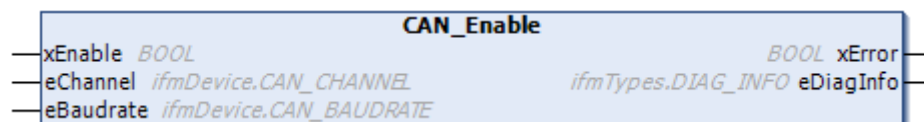
27160

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

27160



If the CAN interface is set as interface for service communication (in the file comconf.cfg entry Service=TRUE), the baud rate at the FB input eBaudrate must be identical!  
Otherwise: No communication is possible and error message  
ERR\_BAUDRATE\_INVALID\_OR\_ALREADY\_SET

## Description

27181

The FB activates the CAN Layer 2 functions of a CAN interface with a certain transmission rate. Simultaneously the FB writes information about the current state of the CAN interface into the global variable CAN State.

Changes of the transmission rate or of the CAN interface are applied at once. All existing reception and send buffer storages are deleted.



The FB does not have any influence on a CANopen Manager / CANopen Device at the selected CAN interface. In this case the FB cannot change the transmission rate of the CAN interface.

## Input parameters

27265

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Disable function block
			TRUE	Enable function blocks
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. <a href="#">212</a> )	
eBaudrate	CAN_BAUDRATE	Baud rate of the CAN channel	→ <b>CAN_BAUDRATE (ENUM)</b> (→ p. <a href="#">212</a> )	

## Output parameters

27070

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>▪ Error occurred</li> <li>▪ Action could not be executed</li> <li>▪ Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

### Diagnostic codes:

- STAT\_INACTIVE      State: FB/Function is inactive.
- STAT\_DONE      State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_BUS\_OFF      Error: CAN interface is in the "BUS OFF" state
- ERR\_INTERNAL      Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_INVALID\_VALUE      Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_BAUDRATE\_INVALID\_OR\_ALREADY\_SET      Error: The required baud rate cannot be set because it is invalid or a different baud rate has already been selected.
- ERR\_UNDEFINED      Error: Unknown error
  - Contact the ifm Service Center!

## CAN\_Recover

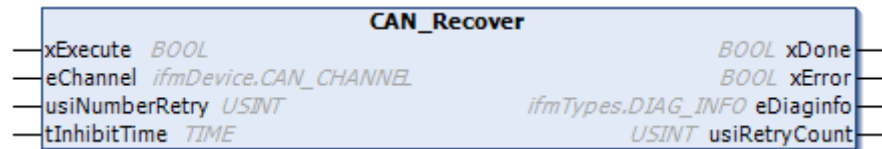
27162

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27186

The FB controls the processing of a failure of the CAN channel.

The call of the FB triggers the following actions:

- If the CAN channel fails the CAN interface is reset and rebooted.
- All buffer storages are emptied.



If the CAN channel keeps failing after the maximum number of recovery attempts has been exceeded, the CAN bus remains in the error state.

- Call FB again to repeat the execution of the recovery function.

## Input parameters

27266

Parameter	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE ⇔ TRUE	FB is executed once
			Other	No impact on FB processing
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. <a href="#">212</a> )	
usiNumberRetry	USINT	Max. number of retries	E.g. 4	
tInhibitTime	TIME	Time until the CAN interface is started again after the detection of a CAN bus failure	E.g. #2ms	

## Output parameters

27305

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	<ul style="list-style-type: none"> <li>▪ FB successfully executed</li> <li>▪ FB can be called again</li> </ul>
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>▪ Error occurred</li> <li>▪ Action could not be executed</li> <li>▪ Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
usiRetryCount	USINT	Counter for retries carried out since the last activation of the FB		

### Diagnostic codes:

- STAT\_INACTIVE                      State: FB/Function is inactive.
- STAT\_DONE                          State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INACTIVE\_INTERFACE      Error: Selected CAN channel is deactivated.
- ERR\_INTERNAL                      Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_INVALID\_VALUE              Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_UNDEFINED                   Error: Unknown error
  - ▶ Contact the ifm Service Center!



## CAN\_RemoteRequest

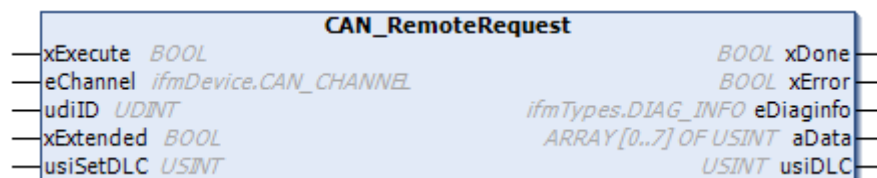
27163

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27184

The FB sends the request for a CAN Remote message into a CAN network. The FB provides the data of the response message in an array. The FB supports standard and extended frames.

## Input parameters

27264

Parameter	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE ⇒ TRUE	FB is executed once
			Other	No impact on FB processing
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. <a href="#">212</a> )	
udiID	UDINT	Identifier of the CAN message	<ul style="list-style-type: none"> <li>for Standard Frame (11 bits identifier): 0 ... 2047</li> <li>for Extended-Frame (29 bits identifier): 0 ... 536.870.911</li> </ul>	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard Frame*
			TRUE	Extended Frame
usiSetDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 8	0 bytes* ... 8 bytes

\* ... preset value

## Output parameters

27072

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	<ul style="list-style-type: none"> <li>▪ FB successfully executed</li> <li>▪ FB can be called again</li> </ul>
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>▪ Error occurred</li> <li>▪ Action could not be executed</li> <li>▪ Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
aData	ARRAY [0...7] OF USINT	Array for storage of the data received		
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 8	0 bytes* ... 8 bytes

### Diagnostic data:

- STAT\_INACTIVE      State: FB/Function is inactive.
- STAT\_DONE      State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_BUSY      State: FB/Function is currently executed.
- ERR\_BUFFER\_OVERFLOW      Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR\_INVALID\_VALUE      Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_INTERNAL      Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_UNDEFINED      Error: Unknown error
  - Contact the ifm Service Center!
- ERR\_INACTIVE\_INTERFACE      Error: Selected CAN channel is deactivated.

## CAN\_RemoteResponse

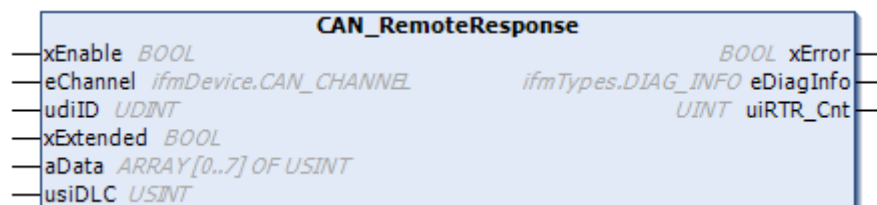
27164

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27188

The FB replies as reaction to the request of a CAN Remote message and sends the data required into a CAN network.

As long as the FB is activated it responds to each remote request message (automatic reply).

Several FB calls are possible during one PLC cycle.

## Input parameters

27263

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Disable function block
			TRUE	Enable function blocks
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. <a href="#">212</a> )	
udiID	UDINT	Identifier of the CAN message	<ul style="list-style-type: none"> <li>for Standard Frame (11 bits identifier): 0 ... 2047</li> <li>for Extended-Frame (29 bits identifier): 0 ... 536.870.911</li> </ul>	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard Frame*
			TRUE	Extended Frame
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0	0 bytes*
			...	...
			8	8 bytes

\* ... preset value

## Output parameters

27304

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
uiRTR_Cnt	UINT	Number of received remote requests after the last FB call		

### Diagnostic code:

- STAT\_INACTIVE                      State: FB/Function is inactive.
- STAT\_DONE                          State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INACTIVE\_INTERFACE      Error: Selected CAN channel is deactivated.
- ERR\_BUFFER\_OVERFLOW          Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR\_INVALID\_VALUE              Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_INTERNAL                      Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_UNDEFINED                   Error: Unknown error
  - Contact the ifm Service Center!

## CAN\_Rx

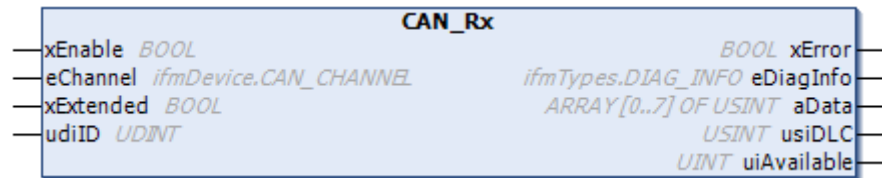
27165

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

60533

The FB receives CAN messages with a defined identifier.

The FB receives all CAN messages with the indicated identifier between 2 FB calls and stores them in a FIFO buffer storage. The number of the received CAN messages is displayed. The CAN message received first is always provided on the output.

If there are several CAN messages in the FIFO buffer storage, the function block can be called until the output is `uiAvailable=0` and all CAN messages have been read from the FIFO buffer storage.



The function block has the following behaviour after changing the receive ID during the runtime of the application

- > The FB memory be not be reset completely. The outputs `aData` and `usiDLC` keep the last values. The `uiAvailable` counter is set to 0.
- Only use the FB with static (unchanged in operation) ID configurations at the inputs.
- Check the value of `uiAvailable`  $\neq$  0 before using the data.

## Input parameters

27267

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Disable function block
			TRUE	Enable function blocks
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. 212)	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard Frame*
			TRUE	Extended Frame
udiID	UDINT	Identifier of the CAN message	<ul style="list-style-type: none"> <li>▪ for Standard Frame (11 bits identifier): 0 ... 2047</li> <li>▪ for Extended-Frame (29 bits identifier): 0 ... 536.870.911</li> </ul>	

## Output parameters

27307

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>▪ Error occurred</li> <li>▪ Action could not be executed</li> <li>▪ Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
aData	ARRAY [0...7] OF USINT	Array for storage of the data received		

usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 8	0 bytes* ... 8 bytes
uiAvailable	UINT	<ul style="list-style-type: none"> <li>Number of received CAN messages since the last FB call</li> <li>Current CAN message is taken into account</li> </ul>	0	No CAN messages received between 2 FB calls
			n	n CAN messages received

#### Error codes:

- STAT\_INACTIVE                      State: FB/Function is inactive.
- STAT\_DONE                         State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INACTIVE\_INTERFACE      Error: Selected CAN channel is deactivated.
- ERR\_BUFFER\_OVERFLOW          Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR\_INVALID\_VALUE              Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_INTERNAL                    Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_UNDEFINED                  Error: Unknown error
  - Contact the ifm Service Center!

## CAN\_RxMask

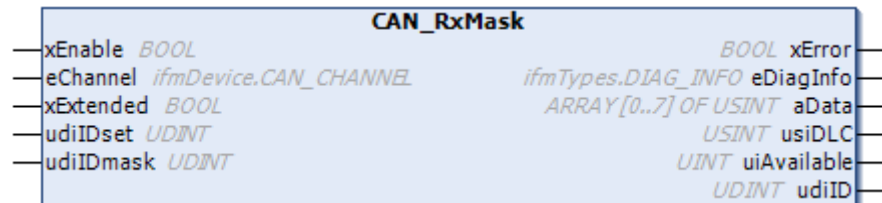
27166

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))



## Description

60536

The FB receives CAN messages of a non-coherent area. The area is defined by a bit pattern and a bit mask.

The following rules apply to the bit mask:

- 0: The equivalent bit of the CAN identifier can be 0 or 1
- 1: The equivalent bit of the CAN identifier must have the same value as the bit in the bit pattern

### Example:

Samples: 000 0010 0000

Mask: 000 1111 1111

Result: xxx 0010 0000

All CAN messages with an identifier whose 8 least significant bits have the value "0010 0000" are received.

e.g. 110 **0010 0000** 000 **0010 0000**, 001 **0010 0000**



General behaviour of the FB: → **CAN\_Rx** (→ p. [225](#))



The function block has the following behaviour after changing the receive ID during the runtime of the application

- > The FB memory be not be reset completely. The outputs aData and usiDLC keep the last values. The uiAvailable counter is set to 0.
- ▶ Only use the FB with static (unchanged in operation) ID configurations at the inputs.
- ▶ Check the value of uiAvailable <> 0 before using the data.

## Input parameters

27268

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Disable function block
			TRUE	Enable function blocks
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. <a href="#">212</a> )	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard Frame*
			TRUE	Extended Frame
udiIDSet	UDINT	Preset bit pattern for the masking of the identifier of the CAN message	E.g. 000 0010 0000	
udiIDMask	UDINT	Bit pattern of the required area 1 ... bit relevant for selection 0 ... bit not relevant for selection	E.g. 000 1111 1111	

\* ... preset value

## Output parameters

27303

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
aData	ARRAY [0...7] OF USINT	Array for storage of the data received		
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 8	0 bytes* ... 8 bytes
uiAvailable	UINT	<ul style="list-style-type: none"> <li>Number of received CAN messages since the last FB call</li> <li>Current CAN message is taken into account</li> </ul>	0	No CAN messages received between 2 FB calls
			n	n CAN messages received
udiID	UDINT	Identifier of the CAN message	<ul style="list-style-type: none"> <li>for Standard Frame (11 bits identifier): 0 ... 2047</li> <li>for Extended-Frame (29 bits identifier): 0 ... 536.870.911</li> </ul>	

### Diagnostic codes:

- STAT\_INACTIVE                      State: FB/Function is inactive.
- STAT\_DONE                         State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INACTIVE\_INTERFACE       Error: Selected CAN channel is deactivated.
- ERR\_BUFFER\_OVERFLOW           Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR\_INVALID\_VALUE              Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_INTERNAL                     Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_UNDEFINED                  Error: Unknown error
  - Contact the ifm Service Center!

## CAN\_RxRange

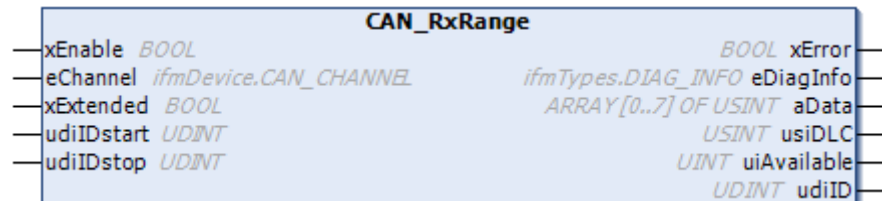
27167

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

60538

The FB receives CAN messages of a contiguous area with standard identifier (11 bit). The area is defined by an upper and lower limit.

The following rules apply to the definition of this area:

- Lower and upper limit: 0 ... 2047 (11 bit identifier)
- The value for the lower limit must be ≤ the value of the upper limit.

Example:

Lower limit 000 0000 0010

Upper limit 000 0000 1000

Result: All CAN messages with an identifier whose 4 least significant bits have a value between "0010" and "1000" are received.



Using the FB CAN\_RxRange for CAN messages with Extended Identifier (29 bit).

> Data loss is possible:

- ▶ Use the FB CAN\_RxRange only for CAN messages with standard identifier (11 bit).
- ▶ Set the block input xExtended = FALSE (standard setting).
- ▶ Use the following FB for CAN messages with Extended Identifier (29 bit): → **CAN\_RxRangeExt** (→ p. [234](#))



General behaviour of the FB: → **CAN\_Rx** (→ p. [225](#))



The function block has the following behaviour after changing the receive ID during the runtime of the application

- > The FB memory be not be reset completely. The outputs aData and usiDLC keep the last values. The uiAvailable counter is set to 0.
- ▶ Only use the FB with static (unchanged in operation) ID configurations at the inputs.
- ▶ Check the value of uiAvailable <> 0 before using the data.

## Input parameters

60539

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Disable function block
			TRUE	Enable function blocks
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. 212)	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bit identifier) - Extended Frame (29 bit identifier)	FALSE	Standard Frame* Required setting. Do not change!
			TRUE	Extended Frame: Do not use! May lead to data loss!
udiIDStart	UDINT	Start of the required area	E.g. 000 0000 0010	
udiIDStop	UDINT	End of the required area	E.g. 000 0000 1000	

\* ... preset value

## Output parameters

60541

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
aData	ARRAY [0...7] OF USINT	Array for storage of the data received		
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 8	0 bytes* ... 8 bytes
uiAvailable	UINT	<ul style="list-style-type: none"> <li>Number of received CAN messages since the last FB call</li> <li>Current CAN message is taken into account</li> </ul>	0	No CAN messages received between 2 FB calls
			n	n CAN messages received
udiID	UDINT	Identifier of the CAN message	Standard Frame (11 Bit Identifier): 0 ... 2047	

**Diagnostic codes:**

- STAT\_INACTIVE                      State: FB/Function is inactive.
- STAT\_DONE                         State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INACTIVE\_INTERFACE       Error: Selected CAN channel is deactivated.
- ERR\_BUFFER\_OVERFLOW           Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR\_INVALID\_VALUE              Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_INTERNAL                     Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED                   Error: Unknown error
  - ▶ Contact the ifm Service Center!

## CAN\_RxRangeExt

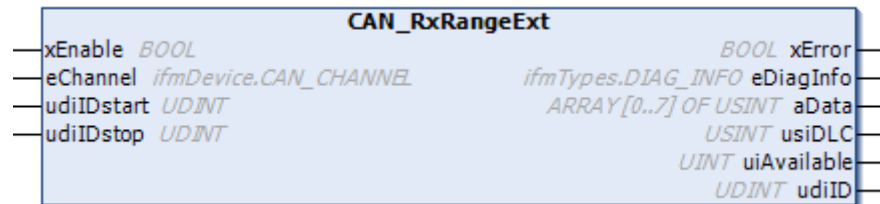
60544

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

60545

The FB receives CAN messages of a contiguous area with Extended Identifier (29 bit). The area is defined by an upper and lower limit.

The following rules apply to the definition of this area:

- Lower and upper limit: 0 ... 536 870 911
- The value for the lower limit must be ≤ the value of the upper limit.

### Example:

Lower limit 0 0000 0000 0000 0000 0000 0000 0010

Upper limit 0 0000 0000 0000 0000 0000 0000 1000

Result: All CAN messages with an identifier whose 4 least significant bits have a value between "0010" and "1000" are received.



General behaviour of the FB: → **CAN\_Rx** (→ p. [225](#))

FB for standard identifier (11 bit): → **CAN\_RxRange** (→ p. [231](#))



The function block has the following behaviour after changing the receive ID during the runtime of the application

- > The FB memory be not be reset completely. The outputs aData and usiDLC keep the last values. The uiAvailable counter is set to 0.
- ▶ Only use the FB with static (unchanged in operation) ID configurations at the inputs.
- ▶ Check the value of uiAvailable <> 0 before using the data.

## Input parameters

60546

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Disable function block
			TRUE	Enable function blocks
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. <a href="#">212</a> )	
udiIDStart	UDINT	Start of the required area	e.g. 0 0000 0000 0000 0000 0000 0010	
udiIDStop	UDINT	End of the required area	e.g. 0 0000 0000 0000 0000 0000 1000	

## Output parameters

60549

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
aData	ARRAY [0...7] OF USINT	Array for storage of the data received		
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 8	0 bytes* ... 8 bytes
uiAvailable	UINT	<ul style="list-style-type: none"> <li>Number of received CAN messages since the last FB call</li> <li>Current CAN message is taken into account</li> </ul>	0	No CAN messages received between 2 FB calls
			n	n CAN messages received
udiID	UDINT	Identifier of the CAN message	<ul style="list-style-type: none"> <li>for Extended-Frame (29 bits identifier): 536.870.911</li> </ul>	

**Diagnostic codes:**

- STAT\_INACTIVE                      State: FB/Function is inactive.
- STAT\_DONE                         State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INACTIVE\_INTERFACE       Error: Selected CAN channel is deactivated.
- ERR\_BUFFER\_OVERFLOW           Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR\_INVALID\_VALUE              Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_INTERNAL                     Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED                  Error: Unknown error
  - ▶ Contact the ifm Service Center!



## CAN\_Status

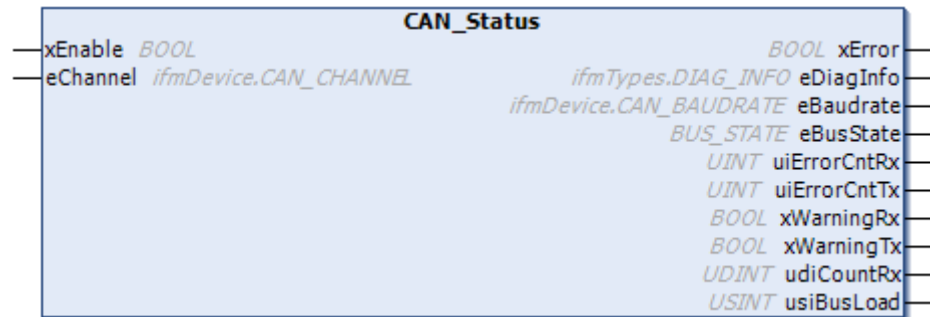
39561

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



## Description

60552

The FB reads the current status of the CAN network and provides the following status and diagnostic information:

- Baud rate
- Status of the CAN bus (status diagram)
- Counter Rx error
- Counter Tx error
- Warning Rx error
- Warning Tx error
- Counter of received CAN messages
- Bus load

If the value range of `udiCountRx` is exceeded, the counter starts at 0.

The bus load is calculated for standard frames. When using Extended Frames, the actual bus load may be higher than indicated on `usiBusLoad`.

## Input parameters

39519

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Disable function block
			TRUE	Enable function blocks
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. <a href="#">212</a> )	

## Output parameters

39577

Parameter	Data type	Description	Possible value	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
eBaudrate	CAN_BAUDRATE	Baud rate of the CAN channel	→ <b>CAN_BAUDRATE (ENUM)</b> (→ p. <a href="#">212</a> )	
eBusState	BUS_STATE	Current state of CAN interface	→ <b>BUS_STATE (ENUM)</b>	
uiErrorCntRx	UINT	Error counter Rx messages	0...65535	
uiErrorCntTx	UINT	Error counter Tx messages	0...65535	
xWarningRx	BOOL	Rx error: Threshold for warning message exceeded (uiErrorCntRx > 96)	FALSE	no warning
			TRUE	warning
xWarningTx	BOOL	Tx error: Threshold for warning message exceeded (uiErrorCntTx > 96)	FALSE	no warning
			TRUE	warning
udiCountRx	UDINT	Number of detected CAN messages (independent of configured Rx messages)	0...4294967295	
usiBusLoad	USINT	Bus load (in percent)	0...100	

### Diagnostic codes:

- **ERR\_INTERNAL**      Error: Internal system error
  - ▶ Contact the ifm Service Center!
- **ERR\_UNDEFINED**      Error: Unknown error
  - ▶ Contact the ifm Service Center!

## CAN\_Tx

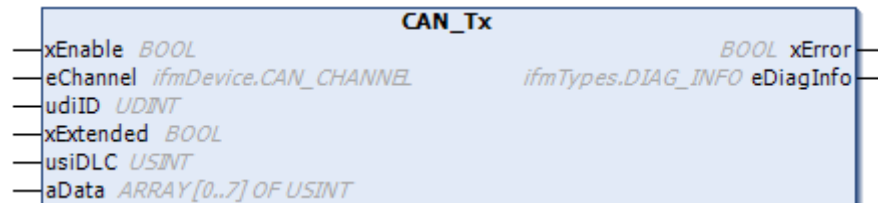
27168

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27183

By means of this FB CAN messages can be sent asynchronously. The FB writes the configured CAN message into the buffer storage of the selected CAN channel. When the CAN message is transmitted depends on the state of the CAN channel and the buffer storage. The FB and the PLC cycle do not have any influence on this.



The FB can be called several times during a PLC cycle.

The repeated call of the FB during a PLC cycle triggers a repeated transmission of the CAN message within the PLC cycle.

## Input parameters

27272

Parameters	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Disable function block
			TRUE	Enable function blocks
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. 212)	
udiID	UDINT	Identifier of the CAN message	<ul style="list-style-type: none"> <li>for Standard Frame (11 bits identifier): 0 ... 2047</li> <li>for Extended-Frame (29 bits identifier): 0 ... 536.870.911</li> </ul>	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard Frame*
			TRUE	Extended Frame
usiDLC	UINT	Number of the data bytes in the CAN message (DLC = Data Length Count)	0 ... 8	0 bytes* ... 8 bytes
aData	ARRAY [0...7] OF USINT	Array with the data to be sent		

\* ... preset value

## Output parameters

27306

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

### Diagnostic codes:

- STAT\_INACTIVE      State: FB/Function is inactive.
- STAT\_DONE      State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INACTIVE\_INTERFACE      Error: Selected CAN channel is deactivated.
- ERR\_BUFFER\_OVERFLOW      Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR\_INVALID\_VALUE      Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_INTERNAL      Error: Internal system error  
► Contact the ifm Service Center!
- ERR\_UNDEFINED      Error: Unknown error  
► Contact the ifm Service Center!

**CAN\_Info (GVL)**

27161

Name	Description	Data type	Possible values	
eBusState	Status of the CAN interface to CiA 11898	→ <b>CAN_BUS_STATE (STRUCT)</b> (→ p. 242)	Undefined	
uiBaudRate	Current baud rate	UINT	0*...65535	
udiRxCount	Counter for all messages detected at the CAN interface	UINT	0*...65535	
uiErrorCntRx	Error counter Rx (receive)	UINT	0*...65535	
uiErrorCntTx	Error counter Tx (send)	UINT	0*...65535	
xWarningRx	Warning signal for error counter Rx	BOOL	FALSE*	uiErrorCntRx < 96
			TRUE	uiErrorcntRx ≥ 96
xWarningTx	Warning signal for error counter Tx	BOOL	FALSE*	uiErrorCntRx < 96
			TRUE	uiErrorcntRx ≥ 96

\* = preset value

**CAN\_BUS\_STATE (STRUCT)**

27158

Name	Description	Possible values		Data type	Value
CAN_BUS_STATE	Status of the CAN interface	UNDEFINED	Interface not available or not configured	INT	0
		ERROR_ACTIVE	Error counter Tx/Rx ≤ 127	INT	1
		ERROR_PASSIVE	Error counter Tx/Rx > 127 and Error counter Tx > 255	INT	2
		ERROR_WARNING	Error counter Tx/Rx > 96 and Error counter Tx/Rx ≤ 127	INT	3
		BUS OFF	Error counter Tx > 255	INT	65535

## 12.4.2 Library ifmCANopenManager.library

### Content

COP_GetNodeState .....	244
COP_SDRead .....	246
COP_SDWrite .....	248
COP_SendNMT .....	250
NMT_SERVICE (ENUM) .....	251
NMT_STATES (ENUM) .....	251

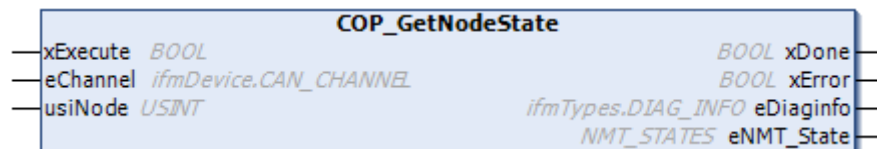
27282

The library contains program blocks (POU) and data structures for the programming of the functionality of a CANopen Manager.

## COP\_GetNodeState

27174

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmCANopenManager.library  
**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### Description

27189

The FB indicates the current state of a CANopen node.

### Input parameters

27209

Parameters	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE	Do not execute FB
			TRUE	Execute FB. Execute the FB (xExecute = TRUE) until the function block execution is successfully terminated (xDone = TRUE). If xDone = TRUE, then reset the input xExecute to FALSE.
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. <a href="#">212</a> )	
usiNode	USINT	ID of the CANopen node	0	Local device
			1 ... 127	ID of the CANopen node

### Output parameters

27074

Parameters	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	<ul style="list-style-type: none"> <li>▪ FB successfully executed</li> <li>▪ FB can be called again</li> </ul>
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed



Parameters	Data type	Description	Possible values
			<div>TRUE</div> <ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)
eNMT_State	NMT_STATES	State of the CANopen node	→ <b>NMT_STATES (ENUM)</b> (→ p. <a href="#">251</a> )

## Diagnostic codes:

- STAT\_INACTIVE                      State: FB/Function is inactive.
- STAT\_BUSY                          State: FB/Function is currently executed.
- STAT\_DONE                          State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INTERNAL                      Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_DEVICE\_NOT\_AVAILABLE       Error: Selected device unknown / not configured
- ERR\_INVALID\_CHANNEL              Error: Selected communication channel unknown / not configured

## COP\_SDOread

27175

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmCANopenManager.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27182

The FB reads the contents of a Service Data Object (SDO) and writes them into a buffer storage. The SDO is selected via the CAN interface, the ID of the CANopen node, as well as index and subindex of the object directory.

The CANopen node has to reply to the request of the FB within a period of time defined by the user.

## Input parameters

27210

Parameters	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE	Do not execute FB
			TRUE	Execute FB. Execute the FB (xExecute = TRUE) until the function block execution is successfully terminated (xDone = TRUE). If xDone = TRUE, then reset the input xExecute to FALSE.
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. <a href="#">212</a> )	
usiNode	USINT	ID of the CANopen node	0	Local device
			1 ... 127	ID of the CANopen node
uiIndex	UINT	Index in the object directory		
usiSubIndex	USINT	Subindex of the index in the object directory		
pData	Pointer to USINT	Pointer on buffer storage		
udiBufLen	UDINT	Size of the buffer storage (in byte)		
tTimeout	TIME	Max. response time	E.g. T#25ms	

## Output parameters

27073

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	<ul style="list-style-type: none"> <li>▪ FB successfully executed</li> <li>▪ FB can be called again</li> </ul>
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>▪ Error occurred</li> <li>▪ Action could not be executed</li> <li>▪ Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
udiLen	UDINT	Number of received bytes		

### Diagnostic codes:

- STAT\_INACTIVE                      State: FB/Function is inactive.
- STAT\_DONE                          State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_CHANNEL              Error: Selected communication channel unknown / not configured
- ERR\_INVALID\_VALUE                Error: At least one input parameter is invalid or outside the value range.
- ERR\_BUFFER\_OVERFLOW              Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR\_INTERNAL                      Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_DEVICE\_NOT\_AVAILABLE      Error: Selected device unknown / not configured
- ERR\_SDO\_IDX\_NOT\_EXIST          Error: Object to be read/written does not exist
- ERR\_SDO\_SUBIDX\_NOT\_EXIST      Error: Subobject to be read/written does not exist
- ERR\_SDO\_UNSUPPORTED\_ACCESS    Error: Read/write access to the selected object is not allowed
- ERR\_SDO\_DATA\_TYPE              Error: Data type of the data to be written does not match the object or is outside the value range

## COP\_SDOwrite

27176

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmCANopenManager.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27190

The FB writes the contents of a Service Data Object (SDO). The SDO is selected via the CAN interface, the ID of the CANopen node, as well as index and subindex of the object directory.

## Input parameters

27208

Parameters	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE	Do not execute FB
			TRUE	Execute FB. Execute the FB (xExecute = TRUE) until the function block execution is successfully terminated (xDone = TRUE). If xDone = TRUE, then reset the input xExecute to FALSE.
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. <a href="#">212</a> )	
usiNode	USINT	ID of the CANopen node	0	Local device
			1 ... 127	ID of the CANopen node
uiIndex	UINT	Index in the object directory		
usiSubIndex	USINT	Subindex of the index in the object directory		
pData	Pointer to USINT	Pointer on buffer storage		
udiLen	UDINT	Number of received bytes		
tTimeout	TIME	Max. response time	E.g. T#25ms	

## Output parameters

27069

Parameters	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	<ul style="list-style-type: none"> <li>▪ FB successfully executed</li> <li>▪ FB can be called again</li> </ul>
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>▪ Error occurred</li> <li>▪ Action could not be executed</li> <li>▪ Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

### Diagnostic codes:

- STAT\_INACTIVE                      State: FB/Function is inactive.
- STAT\_DONE                          State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_CHANNEL              Error: Selected communication channel unknown / not configured
- ERR\_INVALID\_VALUE                Error: At least one input parameter is invalid or outside the value range.
- ERR\_BUFFER\_OVERFLOW              Error: Transmission buffer full; CAN message cannot write to buffer storage and is not transmitted
- ERR\_TIMEOUT                        Error: The maximum permissible execution time was exceeded. The action was not finished.
- ERR\_INTERNAL                        Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_DEVICE\_NOT\_AVAILABLE        Error: Selected device unknown / not configured
- ERR\_SDO\_IDX\_NOT\_EXIST           Error: Object to be read/written does not exist
- ERR\_SDO\_SUBIDX\_NOT\_EXIST       Error: Subobject to be read/written does not exist
- ERR\_SDO\_UNSUPPORTED\_ACCESS    Error: Read/write access to the selected object is not allowed
- ERR\_SDO\_DATA\_TYPE                Error: Data type of the data to be written does not match the object or is outside the value range

## COP\_SendNMT

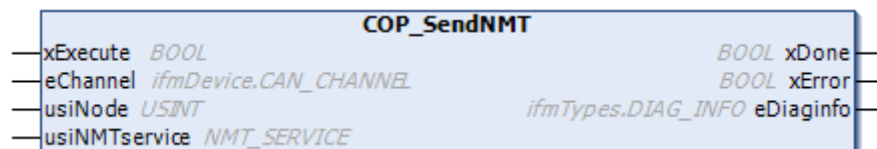
27177

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmCANopenManager.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27180

The FB sends a command for the control of a CANopen node.

## Input parameters

27207

Parameters	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE	Do not execute FB
			TRUE	Execute FB. Execute the FB (xExecute = TRUE) until the function block execution is successfully terminated (xDone = TRUE). If xDone = TRUE, then reset the input xExecute to FALSE.
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ <b>CAN_CHANNEL (ENUM)</b> (→ p. <a href="#">212</a> )	
usiNode	USINT	ID of the CANopen node	0	Local device
			1 ... 127	ID of the CANopen node
usiNMTservice	NMT_SERVICE	Command for the control of a CANopen node	→ <b>NMT_SERVICE (ENUM)</b> (→ p. <a href="#">251</a> )	

## Output parameters

27071

Parameters	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	<ul style="list-style-type: none"> <li>FB successfully executed</li> <li>FB can be called again</li> </ul>

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

## Diagnostic codes:

- STAT\_INACTIVE                      State: FB/Function is inactive.
- STAT\_DONE                          State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_CHANNEL              Error: Selected communication channel unknown / not configured
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_INTERNAL                      Error: Internal system error
  - Contact the ifm Service Center!

**NMT\_SERVICE (ENUM)**

27300

Name	Description	Possible values		Data type	Value
NMT_SERVICE	Command for the control of a CANopen node	SET_PRE_OPERATIONAL	Set preoperational state	INT	1
		SET_OPERATIONAL	Set operational state	INT	2
		RESET_NODE	Reset CAN node	INT	3
		RESET_COMM	Reset communication	INT	4
		STOP_NODE	Stop CAN node	INT	5

**NMT\_STATES (ENUM)**

27301

Name	Description	Possible values		Data type	Value
NMT_STATES	State of the CAN network	INIT	Initialisation	INT	1
		PREOP	Preoperational	INT	2
		OPERATIONAL	Operational	INT	3
		STOP	STOP	INT	4
		NOT_AVAILABLE	Not available	INT	5
		UNKNOWN	Unknown	INT	6

## 12.5 Input and output libraries

### Content

Library ifmConfigSwThreshold.library .....	252
Library ifmFastInput.library .....	256
Library ifmIOcommon.library .....	269
Library ifmIOconfigDiagProt.library .....	289
Library ifmOutGroup .....	298
Library ifmOutHBridge .....	303
Library ifmOutPWM .....	308

27206

### 12.5.1 Library ifmConfigSwThreshold.library

#### Content

ConfigSwThreshold .....	253
-------------------------	-----

27151

The library features program blocks (POU) to configure the switching thresholds of inputs.



## ConfigSwThreshold

27173

**Function block type:** Function block (FB)

**Library:** ifmIOcommon.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

24556

The function block is used to set the switching thresholds for the values TRUE and FALSE of digital input channels in the operating modes IN\_DIGITAL\_CSI and IN\_DIGITAL\_CSO.

- $0 < \text{switching threshold\_L} < \text{switching threshold\_H} < 100\%$
- Default switching threshold\_L: 30% of VBB30
- Default switching threshold\_H: 70% of VBB30

27150



### Information about the function block behaviour:

The signal at the input xExecute must stay set to TRUE until the value at the output xDone becomes = TRUE. Only in this case the function block function will be executed properly.

If the signal at xExecute turns = FALSE before xDone becomes = TRUE, the execution of the function block will be stopped.

- ▶ If xDone becomes = TRUE, then reset xExecute to FALSE.
- > The execution of the function block is stopped and must only be restarted in the event of a fault.

### Programming example:

```
VAR
    xExecute: BOOL:=TRUE;
    ConfigSwThreshold : ifmConfigSwThreshold.ConfigSwThreshold;
END_VAR
ConfigSwThreshold(xExecute:= xExecute )
If ConfigSwThreshold.xDone THEN
    xExecute:=FALSE;
END_IF
```



The diagnostic level of the input will be reset to the standard values when the switching threshold settings are changed. Then, call the function block ConfigDiagLevel.

## Input parameters

24557

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE	Do not execute FB
			TRUE	Execute FB. Execute the FB (xExecute = TRUE) until the function block execution is successfully terminated (xDone = TRUE). After that, reset the input xExecute to FALSE with xDone = TRUE.
uiChannel	UINT	Input channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
uiLowThreshold	UINT	Low switching threshld for digital input in [‰] of VBB30.	0...1000	
uiHighThreshold	UINT	Upper switching threshold for digital input in [‰] of VBB30.	0...1000	



uiLowThreshold and uiHighThreshold are indicated in per mill [‰] of VBB30 in the operating modes IN\_DIGITAL\_CSI and IN\_DIGITAL\_CSO.

## Output parameters

27114

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	<ul style="list-style-type: none"> <li>FB successfully executed</li> <li>FB can be called again</li> </ul>
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	



12.5.2 Library ifmFastInput.library

Content	
FastCount .....	257
IncEncoder .....	260
Period .....	263
COUNT_DIRECTION (ENUM) .....	266
ENCODER_RESOLUTION (ENUM) .....	266
FREQ_SENSE_PERIODS (ENUM) .....	266
MODE_FAST_COUNT (ENUM).....	267
MODE_INC_ENCODER (ENUM) .....	267
MODE_PERIOD (ENUM) .....	268

27283

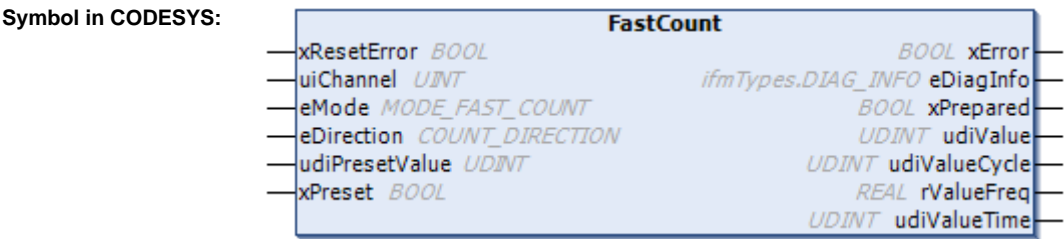
The library contains function blocks (POU) and enumeration types to control the quick inputs of the device.

FastCount

27251

Function block type:      Function block (FB)

Library:                      ifmIFastInput.library



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: →**Using the function blocks** (→ p. [207](#))

Description

27197

The FB functions as a counter block for pulses on fast input channels.

## Input parameters

27278

Parameters	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	Input channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_FAST_COUNT	Operating mode of the input channel	→ <b>MODE_FAST_COUNT (ENUM)</b> (→ p. 267)	
eDirection	COUNT_DIRECTION	Counting direction	→ <b>COUNT_DIRECTION (ENUM)</b> (→ p. 266)	
udiPresetValue	UDINT	Preset counter value	permissible = 0...4 294 967 295	
xPreset	BOOL	Changeover switch: counter function active / adopt preset counter value	FALSE	counter active; the number of counted pulses is issued to udiValue.
			TRUE	The preset counter value is adopted; udiValue = udiPresetValue
udiTimebase	UDINT	Time base for frequency calculation in [ms] Only for eMode = ▪ IN_COUNT_CSI ▪ IN_COUNT_CSO	Value > 0 Example for determining the time base: $\text{udiTimebase} = 2 * 1 / \text{frequency}_{\min}$ Frequency <sub>min</sub> : lowest frequency to be detected	

## Output parameters

27081

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
udiValue	UDINT	Counter value; number of detected pulses	permissible = 0...4 294 967 295	
udiValueCycle	UDINT	Cycle time of the input signal in [μs]		
rValueFreq	REAL	frequency of the input signal in [Hz]		
udiValueTime	UDINT	Elapsed time since the last blanking pulse was detected in [μs]	0...4 294 967 295	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE      State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE      State: FB/Function is inactive.
- ERR\_INTERNAL      Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_UNDEFINED      Error: Unknown error
  - Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION      Error: Internal communication error
- DIAG\_INVALID\_VALUE      At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_ACCESS      FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.

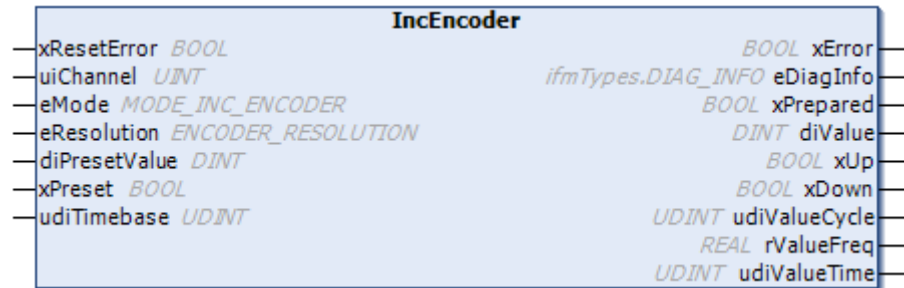
## IncEncoder

27261

**Function block type:** Function block (FB)

**Library:** ifmIFastInput.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27198

The FB is used to configure and to operate a digital input pair to record and count incremental encoder pulses.

Two frequency inputs constitute the input pair (channel A and channel B) that is configured and evaluated via the FB.

### Behaviour at the counter limits

If the applicable value range is exceeded, the output switches to the minimum value of the applicable area. (= overflow)

If the applicable value range is not reached, output switches to the maximum value of the applicable area. (= outside range)



## Input parameters

23300

Parameter	Data type	Meaning	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	1. Input channel (channel A) of the pair of input channels	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Examples:	
			703	Group 7 + channel 3
			1203	Group 12 + channel 3
eMode	MODE_INC_ENCODER	Operating mode of the input channel	→ <b>MODE_INC_ENCODER (ENUM)</b> (→ p. 267)	
eResolution	ENCODER_RESOLUTION	Resolution / encoder mode	→ <b>ENCODER_RESOLUTION (ENUM)</b> (→ p. 266)	
diPresetValue	DINT	Preset counter value	-2 147 483 648...2 147 483 647	
xPreset	BOOL	Changeover switch: counter function active / adopt preset counter value	FALSE	counter active; the number of counted pulses is issued to udiValue.
			TRUE	The preset counter value is adopted; udiValue = udiPresetValue
udiTimebase	UDINT	Time base for frequency calculation in [ms] Only for eMode = ▪ IN_INC_ENCODER_CSI ▪ IN_INC_ENCODER_CSO	Value > 0 Example for determining the time base: $\text{udiTimebase} = 2 * 1 / \text{frequency}_{\min}$ Frequency <sub>min</sub> : lowest frequency to be detected	

## Output parameters

27082

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
diValue	DINT	Counter value; number of detected pulses	permissible = -2 147 483 648...2 147 483 647	
xUp	BOOL	Code sequence upwards	FALSE	No count-up since the last call up
			TRUE	Count-up or overflow since that last call-up
xDown	BOOL	Code sequence downwards	FALSE	No count-down since the last call-up
			TRUE	Count-up or underflow since the last call-up
udiValueCycle	UDINT	Cycle time of the input signal in [µs]		
rValueFreq	REAL	frequency of the input signal in [Hz]		
udiValueTime	UDINT	Elapsed time since the last blanking pulse was detected in [µs]	0...4 294 967 295	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE State: FB/Function is inactive.
- ERR\_INTERNAL Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_UNDEFINED Error: Unknown error
  - Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION Error: Internal communication error
- DIAG\_INVALID\_VALUE At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_ACCESS FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.

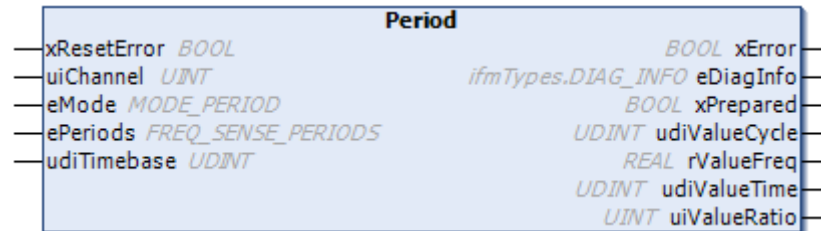
## Period

27311

**Function block type:** Function block (FB)

**Library:** ifmIFastInput.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27199

TheFB is used to configure and to operate an input channel or a pair of input channels to detect and count pulses.

In the operating modes IN\_PHASE\_CSI and IN\_PHASE\_CSO (to be set at the eMode function block input), a phase measurement is carried out on one input channel pair. The input channel pair is defined by indicating the channel with the even number of the input channel pair (channel A) at the input uiChannel.

In the other operating modes, a signal evaluation is carried out at the input channel defined at the uiChannel input.

## Input parameters

27279

Parameters	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	Input channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_PERIOD	Operating mode of the input channel	→ <b>MODE_PERIOD (ENUM)</b>	
ePeriod	FREQ_SENSE_PERIODS	Number of pulse periods for averaging Not for eMode = ▪ IN_FREQUENCY_CSI ▪ IN_FREQUENCY_CSO	→ <b>FREQ_SENSE_PERIODS (ENUM)</b> (→ p. 266)	
udiTimebase	UDINT	Time base for frequency calculation in [ms] Only for eMode = ▪ IN_FREQUENCY_CSI ▪ IN_FREQUENCY_CSO	Value > 0 Example for determining the time base: $\text{udiTimebase} = 2 * 1 / \text{frequency}_{\min}$ Frequency <sub>min</sub> : lowest frequency to be detected	

## Output parameters

27083

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
udiValueCycle	UDINT	Cycle time of the input signal in [μs]		
rValueFreq	REAL	frequency of the input signal in [Hz]		
udiValueTime	UDINT	Elapsed time since the last blanking pulse was detected in [μs]	0...4 294 967 295	
uiValueRatio	UINT	Depends on the mode that is set in the eMode input. Pulse/pause ratio of the input signal in [%] at: <ul style="list-style-type: none"> <li>IN_PERIOD_RATIO_CSI</li> <li>IN_PERIOD_RATIO_CSO</li> <li>IN_PERIOD_RATIO_US_CSI</li> <li>IN_PERIOD_RATIO_US_CSO</li> </ul>		
		Phase shift of the input signal at the B channel to the signal at the A channel in [°] <ul style="list-style-type: none"> <li>IN_PHASE_CSI</li> <li>IN_PHASE_CSO</li> </ul>		

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE      State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE      State: FB/Function is inactive.
- ERR\_INTERNAL      Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED      Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION      Error: Internal communication error
- DIAG\_INVALID\_VALUE      At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_ACCESS      FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.

**COUNT\_DIRECTION (ENUM)**

27178

Name	Description	Possible values	
COUNT_DIRECTION	Counting direction	COUNT_OFF	Counting function off
		COUNT_UP	Counting function up
		COUNT_DOWN	Counting function down

**ENCODER\_RESOLUTION (ENUM)**

27250

Name	Description	Possible values	
ENCODER_RESOLUTION	Resolution	FULL_PERIOD	Counts each rising edge on one channel (A)
		HALF_PERIOD	Counts each rising and falling edge on one channel (A)
		EVERY_EDGE	Counts each rising and falling edge on all channels (A and B)

**FREQ\_SENSE\_PERIODS (ENUM)**

23270

Name	Description	Possible values	
FREQ_SENSE_PERIODS	Number of clock periods for the averaging	PERIODS_n with n = 1	No averaging
		PERIODS_n with n = 2...16	Averaging via n periods After the first full period, the measured value is already provided. After the second full period, averaged values are provided.

**MODE\_FAST\_COUNT (ENUM)**

27292

Name	Description	Possible values	
MODE_FAST_COUNT	Operating mode of the inputs	UNCHANGED	Setting remains unchanged
		IN_COUNT_CSI	Input to count fast signal edges; CSI
		IN_COUNT_CSO	Input to count fast signal edges; CSO
		MONITOR	<ul style="list-style-type: none"> <li>Only output data will be updated.</li> <li>Values, configurations and process data are not written.</li> <li>For applications that are not owners of the resource.</li> </ul>
		OFF	FB deactivated.

**MODE\_INC\_ENCODER (ENUM)**

27293

Name	Description	Possible values	
MODE_INC_ENCODER	Operating mode of the input	UNCHANGED	Setting remains unchanged
		IN_INC_ENCODER_CSI	Input for the evaluation of an incremental encoder, channel A; CSI
		IN_INC_ENCODER_CSO	Input for the evaluation of an incremental encoder, channel A; CSO
		MONITOR	<ul style="list-style-type: none"> <li>Only output data will be updated.</li> <li>Values, configurations and process data are not written.</li> <li>For applications that are not owners of the resource.</li> </ul>
		OFF	FB deactivated.

**MODE\_PERIOD (ENUM)**

60554

Name	Description	Possible values	
MODE_PERIOD	Operating mode of the period input	UNCHANGED	Setting remains unchanged
		IN_FREQUENCY_CSI	Input for frequency measurement; CSI
		IN_FREQUENCY_CSO	Input for frequency measurement; CSO
		IN_PERIOD_RATIO_CSI	Input for absolute and ratiometric period measurement; Measuring resolution 6.25 $\mu$ s; maximum period duration 104 s; CSI
		IN_PERIOD_RATIO_CSO	Input for absolute and ratiometric period measurement; Measuring resolution 6.25 $\mu$ s; maximum period duration 104 s; CSO
		IN_PHASE_CSI	Input pair for phase measurement, CSI
		IN_PHASE_CSO	Input pair for phase measurement, CSO
		OFF	POU disabled.
		IN_PERIOD_RATIO_US_CSI	Input for absolute and ratiometric period measurement; Measuring resolution 1 $\mu$ s; maximum period duration 16 s; CSI
		IN_PERIOD_RATIO_US_CSO	Input for absolute and ratiometric period measurement; Measuring resolution 1 $\mu$ s; maximum period duration 16 s; CSO
		Monitor	<ul style="list-style-type: none"> <li>Only output data will be updated.</li> <li>Values, configurations and process data will not be written.</li> <li>For applications that are not owners of the resource.</li> </ul>



### 12.5.3 Library ifmIOcommon.library

Content	
Input.....	270
Output.....	273
SetLED .....	276
SupplySwitch .....	278
SystemSupply.....	280
Temperature .....	282
FILTER_INPUT (ENUM) .....	284
FILTER_OUTPUT (ENUM) .....	285
MODE_INPUT (ENUM).....	286
MODE_OUTPUT (ENUM).....	287
MODE_INPUT (ENUM).....	287
MODE_SYSTEM_SUPPLY (ENUM).....	288
MODE_TEMPERATURE (ENUM) .....	288
SYS_VOLTAGE_CHANNEL (ENUM) .....	288

27153

The library contains program blocks (POU) and enumeration types for the control of the inputs and outputs of the device.

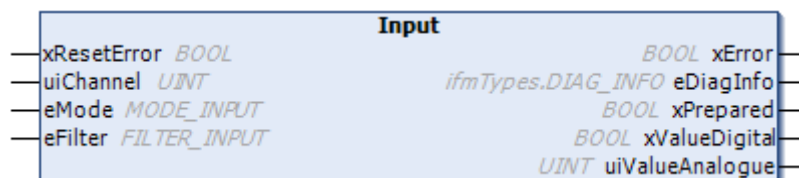
## Input

27262

**Function block type:** Function block (FB)

**Library:** ifmIOcommon.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27116

The FB is used to configure and read a digital or analogue input channel.

### Filter:

The input signal can be changed with a digital low-pass filter.

Configure the filter via the input eFilter.

## Input parameters

27273

Parameters	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE ⇔ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	Input channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. <a href="#">39</a> )	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_INPUT	Operating mode of the input channel	→ <b>MODE_INPUT (ENUM)</b> (→ p. <a href="#">286</a> )	
eFilter	FILTER_INPUT	Filter definition of the input channel	→ <b>FILTER_INPUT (ENUM)</b> (→ p. <a href="#">284</a> )	

## Output parameters

27076

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
xValueDigital	BOOL	Logical state of the input in digital operating mode	FALSE	Low Level or operating mode analogue
			TRUE	High Level
uiValueAnalogue	UINT	Measured input value in analogue operating mode. The interpretation of the input value depends on the setting at the eMode input. <b>MODE_INPUT (ENUM)</b> (→ p. <a href="#">286</a> )	0...65535	

Diagnostic codes (→**Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING                      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE                            State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE                      State: FB/Function is inactive.
- ERR\_INTERNAL                        Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED                      Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION    Error: Internal communication error
- ERR\_COMPARE\_MISMATCH            Error: Deviation of measured value and monitoring value too great
- ERR\_OVERLOAD\_CURRENT            Error: Maximum current exceeded.
- ERR\_STUCK\_AT\_HIGH                Error: Signal frozen, signal state high.
- ERR\_STUCK\_AT\_LOW                 Error: Signal frozen, signal state low.
- ERR\_RANGE\_OVERRUN                Error: Measured value exceeds the set upper limit value.
- ERR\_RANGE\_UNDERRUN              Error: Measured value does not reach the set lower limit value.
- DIAG\_INVALID\_VALUE                At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB                      The selected resource has no valid calibration.  
The displayed values are maybe faulty.
- DIAG\_ACCESS                        FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_OVERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG\_OVERVOLTAGE                 Maximum signal voltage exceeded.
- DIAG\_RANGE\_OVERRUN              The measured value exceeds the set upper limit value.
- DIAG\_RANGE\_UNDERRUN             The measured value does not reach the set lower limit value.
- DIAG\_INVALID\_FILTER                One or several filter settings are invalid.

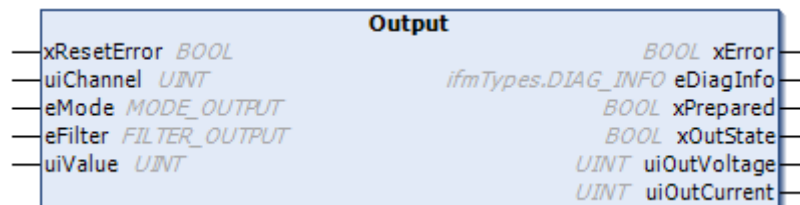
## Output

27302

**Function block type:** Function block (FB)

**Library:** ifmIOcommon.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27192

The FB is used to configure and control a digital or analogue output channel.


## Input parameters

27274

Parameter	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	Output channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. <a href="#">39</a> )	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_OUTPUT	Operating type of the output channel	→ <b>MODE_OUTPUT (ENUM)</b> (→ p. <a href="#">287</a> )	
eFilter	FILTER_OUTPUT	Filter definition of the output channel	→ <b>FILTER_OUTPUT (ENUM)</b> (→ p. <a href="#">285</a> )	
uiValue	UINT	Value that is to be written to the output		
		In the digital mode or sensor supply mode; if setting at the eMode input =	0	Output deactivated
		<ul style="list-style-type: none"> <li>OUT_DIGITAL_CSI</li> <li>OUT_DIGITAL_CSO</li> <li>OUT_SENSOR_05</li> <li>OUT_SENSOR_10</li> </ul>	1	Output activated
		In analogue mode; if setting at the eMode input =	0...10 000	
		<ul style="list-style-type: none"> <li>OUT_ANALOGUE_10</li> </ul> Values indicated in [mV]		

## Output parameters

27077

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
xOutState	BOOL	Return value activation state of the selected output  The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output is deactivated
			TRUE	Output is activated
uiOutVoltage	UINT	Current output voltage (value in mV) Only available for the operating modes "analogue" and "sensor"	0	Operating mode neither "analogue" nor "sensor"
			≠ 0	Output voltage
uiOutCurrent	UINT	Present output current (value in mA) Not available in the operating modes: <ul style="list-style-type: none"> <li>OUT_DIGITAL_CSI</li> <li>OUT_ANALOGUE_10</li> <li>OUT_SENSOR_05</li> <li>OUT_SENSOR_10</li> </ul>	0...final value of the measuring range	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING                      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE                            State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE                      State: FB/Function is inactive.
- ERR\_INTERNAL                        Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED                      Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION    Error: Internal communication error
- ERR\_STUCK\_AT\_HIGH                Error: Signal frozen, signal state high.
- ERR\_STUCK\_AT\_LOW                 Error: Signal frozen, signal state low.
- ERR\_OVERLOAD\_CURRENT            Error: Maximum current exceeded.
- ERR\_LOW\_CURRENT                  Error: Minimum current not reached.
- DIAG\_INVALID\_VALUE                At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB                     The selected resource has no valid calibration.  
The displayed values are maybe faulty.
- DIAG\_ACCESS                        FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_STUCK\_AT\_HIGH               Error: Signal frozen, signal state high.
- DIAG\_STUCK\_AT\_LOW                Error: Signal frozen, signal state low.
- DIAG\_OVERLOAD\_CURRENT           Maximum current exceeded.
- DIAG\_LOW\_CURRENT                 Minimum current not reached.
- DIAG\_INVALID\_FILTER               One or several filter settings are invalid.
- DIAG\_UNDERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs: Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.

SetLED

27315

Function block type: Function block (FB)

Library: ifmIOcommon.library

Symbol in CODESYS:



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

Description

27193

The FB is used to configure and control an LED.

Input parameters

27275

Parameters	Data type	Description	Possible values	
uiChannel	UINT	Output channel of the ED	0...3	Device LED APP 0...3
eColour1	LED_COLOUR (ENUM)	LED colour status 1	→ <b>LED_COLOUR (ENUM)</b> (→ p. <a href="#">213</a> )	
eColour2	LED_COLOUR (ENUM)	LED colour status 0	→ <b>LED_COLOUR (ENUM)</b> (→ p. <a href="#">213</a> )	
eFrequency	ENUM	Flashing frequency of the status LED	→ <b>LED_FLASH_FREQ (ENUM)</b> (→ p. <a href="#">213</a> )	
xOn	BOOL	Switch LED	FALSE	LED off
			TRUE	LED on



## Output parameters

27078

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_DONE                      State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_PREPARING              State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- ERR\_INVALID\_FREQUENCY      Error: Unsupported frequency.
- ERR\_INVALID\_COLOUR        Error: Unsupported colour.
- ERR\_INVALID\_VALUE          Error: At least one input parameter is invalid or outside the value range.
- ERR\_INSTANCE                Error: Instance is ZERO or invalid.
- ERR\_ACCESS                  Error: FB/Funktion cannot access the required resource; Resource is blocked by another task.
- ERR\_UNDEFINED               Error: Unknown error
  - Contact the ifm Service Center!
- ERR\_NOT\_SUPPORTED          Error: Invalid function calls; Function is not supported.

# SupplySwitch

27348

Function block type:      Function block (FB)

Library:                      ifmIOcommon.library

Symbol in CODESYS:



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: →**Using the function blocks** (→ p. [207](#))

## Description

27196

The FB stops all running applications and switches off the voltage supply latching (terminal 30) in order to shut down the device safely.

The voltage supply latching is only deactivated if the following conditions are met:

- Voltage VBB15 < 5.5 V (undervoltage)

The separation from the VBB30 takes place when all IEC tasks are finished.

## Input parameters

27249

Parameter	Data type	Description	Possible values	
eMode	MODE_SUPPLY_SWITCH	Operating mode	→ <b>MODE_INPUT (ENUM)</b> (→ p. <a href="#">287</a> )	
xValue	BOOL	Activate / deactivate latching switch of the device	FALSE	Request deactivation of the latching switch
			TRUE	Activate latching switch

## Output parameters

27075

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE      State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE      State: FB/Function is inactive.
- ERR\_INTERNAL      Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED      Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION      Error: Internal communication error
- DIAG\_ACCESS      FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.

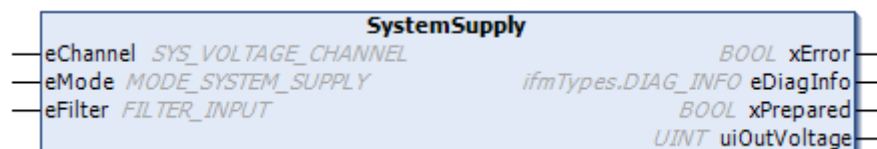
## SystemSupply

27353

**Function block type:** Function block (FB)

**Library:** ifmIOcommon.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27194

The FB indicates the value of the system voltage.

## Input parameters

54533

Parameter	Data type	Meaning	Possible values
eChannel	SYS_VOLTAGE_CHANNEL (ENUM)	System voltage channel	→ <b>SYS_VOLTAGE_CHANNEL (ENUM)</b> (→ p. <a href="#">288</a> )
eMode	MODE_SYSTEM_SUPPLY	Operating mode	→ <b>MODE_SYSTEM_SUPPLY (ENUM)</b> (→ p. <a href="#">288</a> )
eFilter	FILTER_INPUT	Filter definition of the input channel	→ <b>FILTER_INPUT (ENUM)</b> (→ p. <a href="#">284</a> )

## Output parameters

27079

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
uiOutVoltage	UINT	Current output voltage of the selected system voltage channel (value in mV)	0...maximum operating voltage	

Diagnostic codes (→**Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_DONE State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_PREPARING State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_INACTIVE State: FB/Function is inactive.
- ERR\_UNDEFINED Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION Error: Internal communication error
- DIAG\_ACCESS FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_INVALID\_VALUE At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB The selected resource has no valid calibration.  
The displayed values are maybe faulty.
- DIAG\_OVERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs. Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- DIAG\_INVALID\_FILTER One or several filter settings are invalid.

# Temperature


27354

Function block type: Function block (FB)

Library: ifmIOcommon.library

Symbol in CODESYS:



 Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: →**Using the function blocks** (→ p. [207](#))

## Description

54535

The function block indicates the temperature depending on the set channel:

Channel 0 = PCB temperature

Channel 1 = processor temperature

## Input parameters

23246

Parameter	Data type	Meaning	Possible values
uiChannel	UINT	Input channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. <a href="#">39</a> )
			Examples:
			403      Group 4 + channel 3
			502      Group 5 + channel 2
eMode	MODE_TEMPERATURE	Operating mode	→ <b>MODE_TEMPERATURE (ENUM)</b> (→ p. <a href="#">288</a> )
eFilter	FILTER_INPUT	Filter definition of the input channel	→ <b>FILTER_INPUT (ENUM)</b> (→ p. <a href="#">284</a> )

## Output parameters

27080

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
iTemperatureC	INT	Measured temperature (value in °C)	e.g. 35	
iTemperatureF	INT	Measured temperature (value in °F)	e.g. 95	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE      State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE      State: FB/Function is inactive.
- ERR\_UNDEFINED      Error: Unknown error
  - Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION      Error: Internal communication error
- DIAG\_INVALID\_VALUE      At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_ACCESS      FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_LOW\_TEMPERATURE      Temperature in the device does not reach the lower warning limit value.
- DIAG\_HIGH\_TEMPERATURE      Temperature in the device exceeds the upper warning limit value.
- DIAG\_INVALID\_FILTER      One or several filter settings are invalid.

**FILTER\_INPUT (ENUM)**

23166

The input signal can be changed with a digital low-pass filter.

For the output signal of the function bloc, the delay time is changed to the input signal change by the filter. This applies both to the switch-on and the switch-off pulse.

Name	Description	Possible values	Digital signal delay	Analogue signal delay
FILTER_INPUT	Valid filters for inputs of the function blocks	UNCHANGED	No change of settings	
		FILTER_0 (default)	0.6 ms (no digital low-pass filter is set)	1.7 ms (no digital low-pass filter is set)
		FILTER_1	0.9 ms	3.3 ms
		FILTER_2	2.1 ms	7.0 ms
		FILTER_3	4.0 ms	14.1 ms
		FILTER_4	7.6 ms	28.9 ms
		FILTER_5	15.2 ms	58.4 ms
		FILTER_6	30.8 ms	117.2 ms
		FILTER_7	61.6 ms	235.2 ms
		FILTER_8	123.2 ms	470.8 ms
		FILTER_9	246.4 ms	942.4 ms
		FILTER_10	493.2 ms	1885.6 ms
		FILTER_11	986.4 ms	3772.0 ms
		FILTER_12	1972.4 ms	7544.4 ms

Further information → **Filter** (→ p. [476](#))



**FILTER\_OUTPUT (ENUM)**

23167

Filter setting for current measurement of an output.

The signal of the current measurement is damped via a first-order low-pass filter.

Name	Description	Possible values	
FILTER_OUTPUT	Valid filters for outputs of the function blocks	UNCHANGED	No change of settings
		FILTER_0 (default)	1.7 ms
		FILTER_1	1.8 ms
		FILTER_2	2.4 ms
		FILTER_3	3.9 ms
		FILTER_4	7.4 ms
		FILTER_5	14.7 ms
		FILTER_6	29.3 ms
		FILTER_7	58.8 ms
		FILTER_8	117.7 ms
		FILTER_9	235.6 ms
		FILTER_10	471.4 ms
		FILTER_11	943.0 ms
		FILTER_12	1886.1 ms

Further information → **Filter** (→ p. [476](#))

**MODE\_INPUT (ENUM)**

27294

Name	Description	Possible values	
MODE_INPUT	Operating mode of the inputs	UNCHANGED	Preset mode is maintained
		IN_DIGITAL_CSI (default)	Input for analogue value measurement and digital evaluation without diagnostics; CSI
		IN_DIGITAL_CSI_NAMUR	Input for analogue value measurement and digital evaluation with NAMUR-capable diagnostics; CSI
		IN_VOLTAGE_10	Input for analogue current measurement 0...10 V; CSI
		IN_VOLTAGE_32	Input for analogue current measurement 0...32 V; CSI
		IN_VOLTAGE_RATIO	Input for ratiometric current measurement in relation to VBB30; CSI
		IN_CURRENT_CSI	Input for current measurement 0...20 mA; CSI
		IN_RESISTOR	Input for resistance measurement; CSO
		IN_DIGITAL_CSO	Input for analogue value measurement and digital evaluation without diagnostics; CSO
		MONITOR	No parameters or process data are written. Only the FB output data is updated.  For use in a PLC application to which the resource does not belong.
		OFF	FB deactivated.

**MODE\_OUTPUT (ENUM)**

27295

Name	Description	Possible values	
MODE_OUTPUT	Operating mode of the outputs	UNCHANGED	Preset mode is maintained
		OUT_DIGITAL_CSI	Digital output without diagnostics; CSI
		OUT_DIGITAL_CSO (default for standard outputs)	Digital output without diagnostics; CSO
		OUT_ANALOGUE_10	Analogue output to generate a selectable voltage 0...10 V without diagnostics. Generated with the help of a filtered PWM signal. CSO
		OUT_SENSOR_05	Output with fixed output voltage 5 V for the sensor supply without diagnostics and without protection. CSO
		OUT_SENSOR_10	Output with fixed output voltage 10 V for the sensor supply without diagnostics and without protection. CSO
		MONITOR	No parameters or process data are written. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.
		OFF (default for analogue outputs and sensor supply)	FB deactivated.

**MODE\_INPUT (ENUM)**

54537

Name	Description	Possible values	
MODE_SUPPLY_SWITCH	Operating mode	UNCHANGED	Preset mode is maintained
		SYS_SUPPLY_SWITCH	POU enabled.
		MONITOR	No writing of parameters or process data. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.
		OFF	POU disabled.

**MODE\_SYSTEM\_SUPPLY (ENUM)**

54538

Name	Description	Possible values	
MODE_SYSTEM_SUPPLY	Operating mode	UNCHANGED	Preset mode is maintained
		SYS_SUPPLY	POU enabled.
		MONITOR	No parameters or process data are written. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.
		OFF	POU disabled.

**MODE\_TEMPERATURE (ENUM)**

54539

Name	Description	Possible values	
MODE_TEMPERATURE	Operating mode	UNCHANGED	Preset mode is maintained
		SYS_TEMPERATURE	POU enabled.
		MONITOR	No parameters or process data are written. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.
		OFF	POU disabled.

**SYS\_VOLTAGE\_CHANNEL (ENUM)**

27350

Name	Description	Possible values	
SYS_VOLTAGE_CHANNEL	List of all available system voltages.	VBB30	Terminal 30 system voltage
		VBB15	Terminal 15 system voltage of the ignition switch

### 12.5.4 Library ifmIOconfigDiagProt.library

<b>Content</b>	
ConfigDiagLevel .....	290
ConfigDiagProt .....	294
DEST (ENUM) .....	297
eDIAG_PROT_MODE (ENUM) .....	297

27154

The library contains program blocks (POU) and enumeration types to configure the I/O-related diagnostics and protective functions. The behaviour in case of errors and for diagnostic purposes can be set with it.

## ConfigDiagLevel

27171

**Function block type:** Function block (FB)  
**Behaviour model:** Execute  
**Library:** ifmIOconfigDiagProt.library  
**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

23854

The FB is used to set the diagnostic level of the system-internal error detection of input and output channels and output channel groups. You can set, for example, the time delay and the upper and lower limit value of the error detection.



### WARNING

Faulty configuration of the following parameters:

- uiDetectionTime
- uiDiagLimMin / uiDiagLimitMin
- uiDiagLimMax / uiDiagLimitMax

When setting a time behaviour configuration of the lower / upper limit value of input channels/output channels that is different from the factory setting.

- > Risk of personal injuries and/or damage to property.
- > Failure or delay of the safety function is possible.
- ▶ Set upper / lower limit value and check with function test.
- ▶ The uiDetectionTime has an impact on the safety time. The fault will not be detected before the uiDetectionTime has expired.
- ▶ Calculating the safety time.
- ▶ Ensure that the safety time is sufficient to comply with the process safety time ( → **The process safety time** (→ p. [30](#)))



To prevent incorrect diagnostics on outputs in the operating mode OUT\_PWM\_CS0:

- ▶ Set the DetectionTime as follows: uiDetectionTime  $\geq 2 / \text{uiFrequency}$
- ▶ Note: → **PWM1000** (→ p. [312](#)) and → **SF\_PWM1000** (→ p. [367](#)).

**Information about the function block behaviour:**

The signal at the input xExecute must stay set to TRUE until the value at the output xDone becomes = TRUE. Only in this case the function block function will be executed properly.

If the signal at xExecute turns = FALSE before xDone becomes = TRUE, the execution of the function block will be stopped.

- ▶ If xDone becomes = TRUE, then reset xExecute to FALSE.
- > The execution of the function block is stopped and must only be restarted in the event of a fault.

**Programming example:**

```
VAR
    xExecute: BOOL:=TRUE;
    ConfigDiagLevel : ifmIOconfigDiagProt.ConfigDiagLevel;
END_VAR
    ConfigDiagLevel(xExecute:= xExecute )
    If ConfigDiagLevel.xDone THEN
        xExecute:=FALSE;
    END_IF
```

## Input parameters

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE	Do not execute FB
			TRUE	Execute FB. Execute the FB (xExecute = TRUE) until the function block execution is successfully terminated (xDone = TRUE). After that, reset the input xExecute to FALSE with xDone = TRUE.
uiChannel	UINT	Input/output channel or output channel group to be set	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. <a href="#">39</a> )	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eDestination	DEST	Type of channel / output channel group	→ <b>DEST (ENUM)</b> (→ p. <a href="#">297</a> )	
uiDetectionTime	UINT	Time delay in [ms] until an error issued after the user-defined upper/lower limit has been exceeded or not reached.		
uiDiagLimMin	UINT	Lower limit value to diagnose an error. For inputs: Indication in [ $\mu$ A], [mV], [ $\Omega$ ] or [%], depends on the operating mode that is set. For outputs: Indication in [mA] or [mV], depending on the operating mode that is set.		
uiDiagLimMax	UINT	Upper limit value to diagnose an error. For inputs: Indication in [ $\mu$ A], [mV], [ $\Omega$ ] or [%], depends on the operating mode that is set. For outputs: Indication in [mA] or [mV], depending on the operating mode that is set.		



uiDiagLimitMin and uiDiagLimitMax are indicated in per mill [%] of VBB30 in the operating modes IN\_DIGITAL\_CSI and IN\_DIGITAL\_CSO.



## Output parameters

27089

Parameters	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	<ul style="list-style-type: none"> <li>▪ FB successfully executed</li> <li>▪ FB can be called again</li> </ul>
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>▪ Error occurred</li> <li>▪ Action could not be executed</li> <li>▪ Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_BUSY State: FB/Function is currently executed.
- STAT\_DONE State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE State: FB/Function is inactive.
- ERR\_INTERNAL Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION Error: Internal communication error
- DIAG\_INVALID\_VALUE At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_ACCESS FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_WINDOW Invalid value for user diagnostics window.

## ConfigDiagProt

27172

**Function block type:** Function block (FB)  
**Behaviour model:** Execute  
**Library:** ifmIOconfigDiagProt.library  
**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27117

The FB is used to configure the diagnostic and protective behaviour of input and output channels and output channel groups.



### WARNING

When configuring the diagnostic and protective behaviour of input channels / output channels in a way that is different from the factory setting (factory setting: eDIAG\_PROT\_MODE = DIAG\_PROT).

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function is possible.
- ▶ Ensure fail-safe diagnostic and protective behaviour of the input channels and output channels.
- ▶ Programming the evaluation of the diagnostic messages / measured values and safe state in the IEC application.

27117



### Information about the function block behaviour:

The signal at the input xExecute must stay set to TRUE until the value at the output xDone becomes = TRUE. Only in this case the function block function will be executed properly. If the signal at xExecute turns = FALSE before xDone becomes = TRUE, the execution of the function block will be stopped.

- ▶ If xDone becomes = TRUE, then reset xExecute to FALSE.
- > The execution of the function block is stopped and must only be restarted in the event of a fault.

**Programming example:**

```

VAR
xExecute: BOOL:=TRUE;
ConfigDiagProt : ifmIOconfigDiagProt.ConfigDiagProt;
END_VAR
ConfigDiagProt(xExecute:= xExecute )
If ConfigDiagProt.xDone THEN
xExecute:=FALSE;
END_IF

```

**Input parameters**

27215

Parameter	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE	Do not execute FB
			TRUE	Execute FB. Execute the FB (xExecute = TRUE) until the function block execution is successfully terminated (xDone = TRUE). After that, reset the input xExecute to FALSE with xDone = TRUE.
uiChannel	UINT	Input/output channel or output channel group to be set	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eDestination	DEST	Type of channel / output channel group	→ <b>DEST (ENUM)</b> (→ p. 297)	
eDiagMode	eDIAG_PROT_MODE	Setting of the error detection type	→ <b>eDIAG_PROT_MODE (ENUM)</b> (→ p. 297)	

## Output parameters

27088

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	<ul style="list-style-type: none"> <li>▪ FB successfully executed</li> <li>▪ FB can be called again</li> </ul>
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>▪ Error occurred</li> <li>▪ Action could not be executed</li> <li>▪ Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_BUSY State: FB/Function is currently executed.
- STAT\_DONE State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE State: FB/Function is inactive.
- ERR\_INTERNAL Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_UNDEFINED Error: Unknown error
  - Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION Error: Internal communication error
- DIAG\_INVALID\_VALUE At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_ACCESS FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_DIAG\_PROT\_MODE Invalid mode for diagnostic/protective function.

**DEST (ENUM)**

27204

Name	Description	Possible values	
DEST	Type of channel	INPUT	Input
		OUTPUT	Output
		OUT_GROUP	Output group

**eDIAG\_PROT\_MODE (ENUM)**

27205

Name	Description	Possible values	
eDIAG_PROT_MODE	Setting of the I/O error diagnostics mode	DIAG_PROT	Initial value Diagnostics and protection activated, in case of error: - ERR message - Set input/output values = 0
		DIAG	Diagnostics activated, protection deactivated, in case of an error: - DIAG message - no error response of the input/output values
		NO_DIAG	Diagnostics and protection deactivated, in case of an error: - No message - No error response of the input/output values

12.5.5 Library ifmOutGroup

Content	
OutputGroup .....	299
FILTER_OUTPUT_GROUP (ENUM) .....	302
MODE_OUTPUT_GROUP (ENUM).....	302

27285

The library contains function blocks (POU) to control extended output functions.

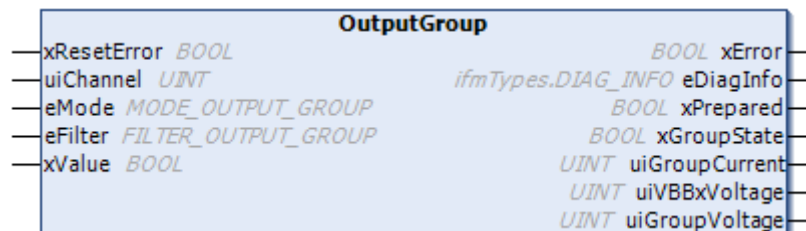
## OutputGroup

27310

**Function block type:** Function block (FB)

**Library:** ifmOutGroup.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27200

The FB controls the activation status of an output group and provides diagnostic information about the group and the connected outputs. Using the FB, an output group including the corresponding outputs can be switched on or off.

## Input parameters

27211

Parameter	Data type	Description	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	Output channel group	→ Data sheet → <b>Note on wiring</b> (→ p. <a href="#">39</a> )	
eMode	MODE_OUTPUT_GROUP	Operating type of the output channel group	→ <b>MODE_OUTPUT_GROUP (ENUM)</b> (→ p. <a href="#">302</a> )	
eFilter	FILTER_OUTPUT_GROUP	Defines the limit frequency of the output filter	→ <b>FILTER_OUTPUT_GROUP (ENUM)</b> (→ p. <a href="#">302</a> )	
xValue	BOOL	Activation requirement for the output group	FALSE	Deactivate output group
			TRUE	Activate output group


27211



The error of an output group will only be reset if all corresponding outputs are error-free.

## Output parameters

23329

Parameter	Data type	Meaning	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
xGroupState	BOOL	Return value activation state of the selected output group  The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output group is deactivated
			TRUE	Output value is activated
uiGroupCurrent	UINT	Measured output current of the entire group in [mA]	available = 0...final value of the measuring range	
uiVBBxVoltage	UINT	Measured voltage before the group switch in [mV]	available = 0...final value of the measuring range	
uiGroupVoltage	UINT	Measured voltage after the group switch in [mV]	available = 0...final value of the measuring range	



Diagnostic codes (→**Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING                      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE                            State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE                      State: FB/Function is inactive.
- ERR\_INTERNAL                        Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED                      Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION    Error: Internal communication error
- ERR\_OVERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs: Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- ERR\_OVERLOAD\_CURRENT            Error: Maximum current exceeded.
- ERR\_GROUP\_SW\_TEST                Error: Start-up test of the group switch has failed.  
Possible causes:
  - VBBx is not supplied.
  - At least one output of the group is supplied externally.
  - At least one output detects STUCK\_AT\_HIGH.
  - Defect at group switch or output
- ERR\_AT\_GROUP\_OUTPUT              Error: At least one output of the output group is in an error state.
- ERR\_STUCK\_AT\_HIGH                Error: Signal frozen, signal state high.
- ERR\_STUCK\_AT\_LOW                 Error: Signal frozen, signal state low.
- ERR\_UNDERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs: Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG\_INVALID\_VALUE                At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB                      The selected resource has no valid calibration.  
The displayed values are maybe faulty.
- DIAG\_ACCESS                        FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_OVERLOAD\_CURRENT            Maximum current exceeded.
- DIAG\_AT\_GROUP\_OUTPUT            At least one output of the output group is in the error state.
- DIAG\_INVALID\_FILTER                One or several filter settings are invalid.
- DIAG\_STUCK\_AT\_HIGH                Error: Signal frozen, signal state high.
- DIAG\_STUCK\_AT\_LOW                Error: Signal frozen, signal state low.

**FILTER\_OUTPUT\_GROUP (ENUM)**

23338

Filter setting for voltage measurement in an output group.

The signal of the voltage measurement is damped via a first-order low-pass filter.

Name	Description	Possible values	
FILTER_OUTPUT_GROUP		UNCHANGED	No change of settings
		FILTER_0	1.7 ms
		FILTER_1	1.8 ms
		FILTER_2	2.4 ms
		FILTER_3	3.9 ms
		FILTER_4	7.4 ms
		FILTER_5	14.7 ms
		FILTER_6	29.3 ms
		FILTER_7	58.8 ms
		FILTER_8	117.7 ms
		FILTER_9	235.6 ms
		FILTER_10	471.4 ms
		FILTER_11	943.0 ms
		FILTER_12	1886.1 ms

Further information → **Filter** (→ p. [476](#))

**MODE\_OUTPUT\_GROUP (ENUM)**

27296

Name	Description	Possible values	
MODE_OUTPUT_GROUP	Operating type of the output group	UNCHANGE	Setting remains unchanged
		OUT_DIGITAL	Digital output without diagnostics and without protection
		MONITOR	No parameters or process data are written. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.
		OFF	POU disabled.

12.5.6    Library ifmOutHBridge

Content	
HBridge.....	304
MODE_BRAKE (ENUM).....	307
MODE_HBRIDGE (ENUM) .....	307

27286

The library contains function blocks (POU) to control extended output functions via an HBridge.

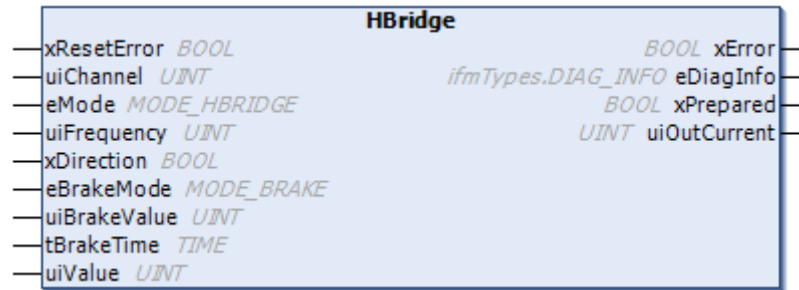
## HBridge

27258

**Function block type:** Function block (FB)

**Library:** ifmIOuHBridge.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

23470

The FB configures and controls a pair of output channels in the "HBridge" operating type to control a motor.



When the function block is called for the first time, both outputs (channel A and channel B) will be tested. The test can take several cycles before the H-bridge can be used.

## Input parameters

23471

Parameter	Data type	Meaning	Possible values	
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	1. Output channel (channel A) of the output channel pair	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. <a href="#">39</a> )	
			Example:	
			106	Group 1 + channel 6 + 7
eMode	MODE_HBRIDGE	Operating mode	→ <b>MODE_HBRIDGE (ENUM)</b> (→ p. <a href="#">307</a> )	
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet	
xDirection	BOOL	The direction in which the current flows via the bridge connections. Determines the direction of rotation of the connected motor.	FALSE	PWM Current Sourcing (CSO) is on channel A
			TRUE	PWM Current Sourcing (CSO) is on channel B
eBrakeMode	MODE_BRAKE	Brake mode that applies when the direction of rotation is changed or when stopping	→ <b>MODE_BRAKE (ENUM)</b> (→ p. <a href="#">307</a> )	
uiBrakeValue	UINT	Pulse/pause ratio of the PWM output signal at the corresponding current sinking output of the bridge in [%] The input is only relevant in the eBrakeModes that end with "_DYNAMIC" (= dynamic brake).	permissible = 0...1 000	
tBrakeTime	TIME	Indicates the braking time for the current sinking side of the bridge The input is only relevant in eBrakeModes ending with "_BTIME".	permissible = 0...1 h	
uiValue	UNIT	Pulse/pause ration of the PWM output signal in [%]	permissible = 0...1000	

## Output parameters

27087

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
uiOutCurrent	UINT	Measured current at the PWM output during normal operation in [mA] When braking, uiOutCurrent is = 0 because no regular current exists in the lowside path.	available = 0...final value of the measuring range	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 463)):

- STAT\_PREPARING      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE      State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE      State: FB/Function is inactive.
- ERR\_INTERNAL      Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED      Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION      Error: Internal communication error
- ERR\_STUCK\_AT\_HIGH      Error: Signal frozen, signal state high.
- ERR\_OVERLOAD\_CURRENT      Error: Maximum current exceeded.
- ERR\_LOW\_CURRENT      Error: Minimum current not reached.
- DIAG\_INVALID\_VALUE      At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB      The selected resource has no valid calibration. The displayed values are maybe faulty.
- DIAG\_ACCESS      FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_STUCK\_AT\_HIGH      Error: Signal frozen, signal state high.
- DIAG\_OVERLOAD\_CURRENT      Maximum current exceeded.
- DIAG\_LOW\_CURRENT      Minimum current not reached.
- DIAG\_UNDERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs: Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG\_INVALID\_FREQUENCY      The frequency is not supported in the active mode.
- DIAG\_INVALID\_DITHER      At least one dither parameter is invalid.

**MODE\_BRAKE (ENUM)**

27290

Name	Description	Possible values	
MODE_BRAKE	Braking mode that is applied when changing the direction (xDirection) or when stopping (uiValue = 0).	UNCHANGED	Setting remains unchanged
		BRAKE_OFF	No braking. The voltage direction is changed immediately.
		BRAKE_EMCY	Emergency brakes: <ul style="list-style-type: none"> <li>In case of change of direction: Braking only during tBrakeTime.</li> <li>When stopping: Braking during and after the tBrakeTime is elapsed.</li> </ul>
		BRAKE_EMCY_BTME	Emergency brakes, but only during tBrakeTime.
		BRAKE_DYNAMIC	Like BRAKE_EMCY mode, but dynamic braking with the uiBrakeValue.
		BRAKE_DYNAMIC_BTME	Like BRAKE_EMCY_BTME mode, but dynamic braking with the uiBrakeValue.

**MODE\_HBRIDGE (ENUM)**

54542

Name	Description	Possible values	
MODE_HBRIDGE	Operating mode	UNCHANGED	Preset mode is maintained
		OUT_H_BRIDGE	POU enabled.
		MONITOR	No parameters or process data are written. Only the FB output data is updated.  For use in a PLC application to which the resource does not belong.
		OFF	POU disabled.

12.5.7 Library ifmOutPWM

Content	
CurrentControl .....	309
PWM1000 .....	312
MODE_CURRENT_CONTROL (ENUM) .....	316
MODE_PWM (ENUM) .....	316

27287

The library function blocks (POU) and enumeration types for pulse width modulation and current control of output channels.



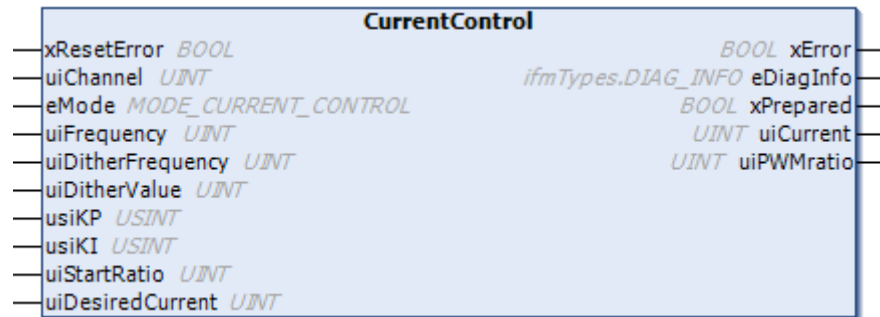
## CurrentControl

27179

**Function block type:** Function block (FB)

**Library:** ifmInOutPWM.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

60557

The FB is used to configure and operate a current controlled output. The current control is provided by means of pulse width modulation (PWM). The configuration of PWM frequency and dither is also done with this FB.

The controller operates in dependence of the period duration of the PWM signal. The two adjustment parameters *usiKI* and *usiKP* represent the integral and the proportional part of the controller.

- It is recommended to set *KI*=50 and *KP*=50 as start values so as to determine the best setting of the controller. Depending on the requested controller behaviour the values can gradually be incremented (controller is stronger / faster) or decremented (controller is weaker / slower).

> With target value *uiDesiredCurrent*=0, the output will be switched off immediately.

The controller has a fast compensation mechanism for voltage drops of the supply voltage. In addition to the controller behaviour of the controller and on the basis of the voltage drop, the ratio of the PWM is increased such that the controller reaches as quickly as possible the desired value.



- When defining the parameter *uiDitherValue*, make sure that the resulting PWM ratio in the operating range of the loop control remains between 0...1000 %:

- $uiPWMratio + uiDitherValue$
- $uiPWMratio - uiDitherValue > 0 \%$ .

With PWM frequencies below 100 Hz plus additional dither, the current control can no longer reach the indicated accuracy (→ data sheet).

## Input parameters

23357

Parameter	Data type	Description	Possible values
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE      Reset request to the lower level system
			else      no action
uiChannel	UINT	Output channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)
			Examples:
			403      Group 4 + channel 3
			502      Group 5 + channel 2
eMode	MODE_ CURRENT_ CONTROL	Operating type of the output channel	→ <b>MODE_CURRENT_CONTROL (ENUM)</b> (→ p. 316)
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet
uiDitherFrequency	UNIT	Frequency for the dither signal at the PWM output in [Hz]	permissible = $0 \dots \text{uiFrequency} / 2$ The value at uiDitherFrequency must be an integer part of the value indicated to uiFrequency. Examples: uiFrequency = 300 Hz uiDitherFrequency = 50 Hz ⇒ $300 / 50 = 6$ ⇒ even factor, valid uiDitherFrequency = 100 Hz ⇒ $300 / 100 = 3$ ⇒ uneven factor, invalid
uiDitherValue	UNIT	Peak-to-peak value of the dither signal which overlays with the PWM signal, in [‰]	permissible = $0 \dots 1000$ If the resulting PWM ratio value is outside the $0 \dots 1000$ ‰ range, the dither value will be temporarily internally reduced to the minimum/maximum value that is possible, so that the mean value of the PWM ratio corresponds with the required value.
usiKP	USINT	Proportional component of the output signal	$0 \dots 255$ If usiKP = 0 then no control.
usiKI	USINT	Integral component of the output signal	$0 \dots 255$ If usiKI = 0 then no control.
uiStartRatio	UNIT	Mark-to-space ratio of the PWM output signal after start of the controller in [‰]	permissible = $0 \dots 1000$ Recommended standard start value: 500
uiDesiredCurrent	UINT	Default value at the output channel. When 0 is set, the output is immediately deactivated.	$0 \dots 65535$

## Output parameters

27086

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
uiCurrent	UINT	Output current signal in [mA]	available = 0...final value of the measuring range	
uiPWMRatio	UINT	PWM pulse ration calculated by the PI controller in [%]		

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 463)):

- STAT\_PREPARING State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE State: FB/Function is inactive.
- ERR\_INTERNAL Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_UNDEFINED Error: Unknown error
  - Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION Error: Internal communication error
- ERR\_STUCK\_AT\_HIGH Error: Signal frozen, signal state high.
- ERR\_OVERLOAD\_CURRENT Error: Maximum current exceeded.
- ERR\_LOW\_CURRENT Error: Minimum current not reached.
- DIAG\_INVALID\_VALUE At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB The selected resource has no valid calibration. The displayed values are maybe faulty.
- DIAG\_ACCESS FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_STUCK\_AT\_HIGH Error: Signal frozen, signal state high.
- DIAG\_OVERLOAD\_CURRENT Maximum current exceeded.
- DIAG\_LOW\_CURRENT Minimum current not reached.
- DIAG\_UNDERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG\_INVALID\_FREQUENCY The frequency is not supported in the active mode.
- DIAG\_INVALID\_DITHER At least one dither parameter is invalid.

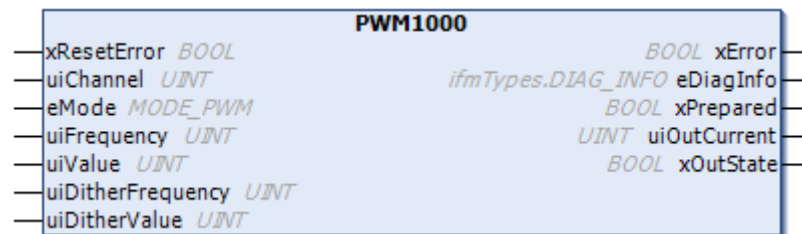
## PWM1000

27313

**Function block type:** Function block (FB)

**Library:** ifmOutPWM.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27201

The FB is used to configure and to operate an output with pulse width modulation.



If the function block ConfigDiagLevel is used:


- Please observe the formula to calculate the DetectionTime → **ConfigDiagLevel** (→ p. [290](#))

## Input parameters

27212

Parameter	Data type	Description	Possible values
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE
			Reset request to the lower level system
uiChannel	UINT	Input channel	else
			no action
			Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)
			Examples:
			403
			Group 4 + channel 3
			502
			Group 5 + channel 2
eMode	MODE_PWM	Operating type of the output channel	→ <b>MODE_PWM (ENUM)</b> (→ p. 316)
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet
uiValue	UNIT	Pulse/pause ration of the PWM output signal in [‰]	permissible = 0...1000
uiDitherFrequency	UNIT	Frequency for the dither signal at the PWM output in [Hz]	permissible = 0...uiFrequency / 2 The value at uiDitherFrequency must be an integer part of the value indicated to uiFrequency. Examples: uiFrequency = 300 Hz uiDitherFrequency = 50 Hz ⇒ 300 / 50 = 6 ⇒ even factor, valid uiDitherFrequency = 100 Hz ⇒ 300 / 100 = 3 ⇒ uneven factor, invalid
uiDitherValue	UNIT	Peak-to-peak value of the dither signal which overlays with the PWM signal, in [‰]	permissible = 0...1000 If the resulting PWM ratio value is outside the 0...1000 ‰ range, the dither value will be temporarily internally reduced to the minimum/maximum value that is possible, so that the mean value of the PWM ratio corresponds with the required value.

## Output parameters

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
uiOutCurrent	UINT	Present output current in [mA]	available = 0...final value of the measuring range	
xOutState	BOOL	Return value activation state of the selected output  The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output is deactivated
			TRUE	Output is activated

Diagnostic codes (→**Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING                      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE                            State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE                      State: FB/Function is inactive.
- ERR\_INTERNAL                        Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED                      Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION    Error: Internal communication error
- ERR\_STUCK\_AT\_HIGH                Error: Signal frozen, signal state high.
- ERR\_OVERLOAD\_CURRENT            Error: Maximum current exceeded.
- ERR\_LOW\_CURRENT                  Error: Minimum current not reached.
- DIAG\_INVALID\_VALUE                At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB                     The selected resource has no valid calibration.  
The displayed values are maybe faulty.
- DIAG\_ACCESS                        FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_STUCK\_AT\_HIGH               Error: Signal frozen, signal state high.
- DIAG\_OVERLOAD\_CURRENT           Maximum current exceeded.
- DIAG\_LOW\_CURRENT                 Minimum current not reached.
- DIAG\_UNDERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs: Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG\_INVALID\_FREQUENCY           The frequency is not supported in the active mode.
- DIAG\_INVALID\_DITHER             At least one dither parameter is invalid.

**MODE\_CURRENT\_CONTROL (ENUM)**

27291

Name	Description	Possible values	
MODE_CURRENT_CONTROL	Operating mode of the output	UNCHANGED	Setting is maintained
		OUT_CURRENT_CSO	Output for current control without diagnostics and without protection; CSO
		MONITOR	No parameters or process data are written. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.
		OFF	FB deactivated.

**MODE\_PWM (ENUM)**

27298

Name	Description	Possible values	
MODE_PWM	Operating mode of the output	UNCHANGED	Setting is maintained
		OUT_PWM_CSI	PWM output without diagnostics; CSI
		OUT_PWM_CSO	PWM output without diagnostics; CSO
		OUT_PWM_CSO_DIAG	PWM output with diagnostics and without protection; CSO
		OUT_PWM_CSO_DIAG_PROT	PWM output with diagnostics and with protection; CSO
		MONITOR	No parameters or process data are written. Only the FB output data is updated. For use in a PLC application to which the resource does not belong.
		OFF	FB deactivated.



## 12.6 Help function libraries

Content	
Library ifmSysInfo.library .....	317

27259

### 12.6.1 Library ifmSysInfo.library

Content	
GetInfo .....	318
aSysInfoList (GVL) .....	320
SYS_INFO (STRUCT) .....	320

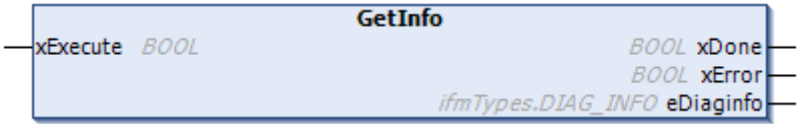
27289

The library contains function block (FB) and data structures (STRUCT, ENUM) for the provision of the device information.

GetInfo

27257

Function block type:     Function block (FB)  
Behaviour model:       EXECUTE  
Library:                ifmSysInfo.library  
Symbol in CODESYS:     



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: →**Using the function blocks** (→ p. [207](#))

Description

27191

The FB reads the following information of the device:

- Firmware version of the device (FW device)
- Firmware version of key row (FW keyboard 1)
- Firmware version of key row (FW keyboard 2)
- Firmware version of key row (FW keyboard 3)
- Firmware version of the watchdog (FW watchdog)
- Firmware version of the I/O driver (FW IO driver)
- Hardware revision (HW revision)
- Serial number of the device (SerialNumber)
- Manufacturing date (Manufacturer Date)

The FB writes the read values in the global variable →**aSysInfoList (GVL)** (→ p. [320](#)).

Input parameters

27270

Parameter	Data type	Description	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE ⇒ TRUE	FB is executed once
			Other	No impact on FB processing

## Output parameters

27309

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	<ul style="list-style-type: none"> <li>FB successfully executed</li> <li>FB can be called again</li> </ul>
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

### Diagnostic codes:

- STAT\_INACTIVE                      State: FB/Function is inactive.
- STAT\_BUSY                          State: FB/Function is currently executed.
- STAT\_DONE                          State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_NOT\_SUPPORTED              Error: Invalid function calls; Function is not supported.
- ERR\_INTERNAL                      Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_INVALID\_VALUE              Error: At least one information type to be read is not supported by the device
- ERR\_UNDEFINED                   Error: Unknown error
  - ▶ Contact the ifm Service Center!

**aSysInfoList (GVL)**

27068

Name	Description	Data type	Value
FIRMWARE_DEVICE	Firmware version of the device	→ <b>SYS_INFO (STRUCT)</b> (→ p. <a href="#">320</a> )	0*
FIRMWARE_KEYBOARD_1	Firmware version of key row 1	→ <b>SYS_INFO (STRUCT)</b> (→ p. <a href="#">320</a> )	0*
FIRMWARE_KEYBOARD_2	Firmware version of key row 2	→ <b>SYS_INFO (STRUCT)</b> (→ p. <a href="#">320</a> )	0*
FIRMWARE_KEYBOARD_3	Firmware version of key row 3	→ <b>SYS_INFO (STRUCT)</b> (→ p. <a href="#">320</a> )	0*
FIRMWARE_WD	Firmware version of the watchdog	→ <b>SYS_INFO (STRUCT)</b> (→ p. <a href="#">320</a> )	0*
FIRMWARE_IO	Firmware version of the I/O driver	→ <b>SYS_INFO (STRUCT)</b> (→ p. <a href="#">320</a> )	0*
HW_REVISION_HW_REL	Hardware revision	→ <b>SYS_INFO (STRUCT)</b> (→ p. <a href="#">320</a> )	0*
DEVICE_SERIAL_NUM	Serial number of the device	→ <b>SYS_INFO (STRUCT)</b> (→ p. <a href="#">320</a> )	0*
MANUFACTURER_DATE	Manufacturing date	→ <b>SYS_INFO (STRUCT)</b> (→ p. <a href="#">320</a> )	0*

\* ... Initialisation value

**SYS\_INFO (STRUCT)**

27349

Designation	Data type	Description	Possible values
sName	STRING (32)	Name of the system component	E.g. firmware version
sValue	STRING (255)	Value of the system component	E.g. 3.1

## 12.7 Safety libraries

### Content

Library ifmIOSafety.library .....	321
Library ifmPLCopenAddonSafe.library .....	374
Library ifmPLCopenSafe.library .....	393

27314

### 12.7.1 Library ifmIOSafety.library

#### Content

SF_Input .....	322
SF_InputBlanking .....	327
SF_OutputEnh .....	332
SF_OutGroupEnh .....	336
SF_PWM1000Enh .....	341
SF_CurrentControlEnh .....	346
SF_HBridgeEnh .....	351
State diagram SF_[Type]Enh .....	356
SF_Output .....	357
SF_OutputGroup .....	360
SF_CurrentControl .....	364
SF_PWM1000 .....	367
SF_HBridge .....	371

27284

The library ifmIOSafety includes validated and certified function blocks as an add-on for standard function blocks from ifm to facilitate their use in the Safety IEC application.

The function of the safety function blocks is not different from the function of the corresponding standard function blocks. However, the safety function blocks additionally provide safety data types and functionalities that are necessary when creating the safety application and to simplify programming in the basic and extended user levels.

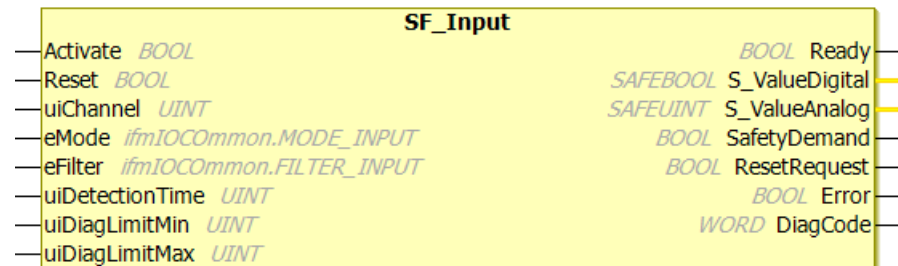
## SF\_Input

27334

**Function block type:** Safety function block (SF)

**Library:** ifmIOSafety.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

24112

The function block SF\_Input is used to read in safe inputs with one channel in digital and analogue modes.

In contrast to the standard function block, this function block only offers the modes that enable 1-channel use.

If error messages occur if Activate=FALSE at the assigned input channel, the messages will be indicated directly if Activate=TRUE. The errors must be handled through the application.


27143

The function block's behaviour is based on the PLCopen safe function blocks according to the PLCopen specification TC5 Safety part 1. The function block features the extended outputs according to the PLCopen specification TC5 Safety part 3, chapter 2.4.

## Input parameters

27241

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error
uiChannel	UINT	Input channel Will be adopted once during the initialisation of the function block.	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. <a href="#">39</a> )	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2

eMode	MODE_INPUT	Operating mode of the input channel Will be adopted once during the initialisation of the function block.	→ <b>MODE_INPUT (ENUM)</b> (→ p. <a href="#">286</a> ) IN_DIGITAL_CSI IN_DIGITAL_CSI_NAMUR IN_VOLTAGE_10 IN_VOLTAGE_32 IN_VOLTAGE_RATIO IN_CURRENT_CSI	
eFilter	FILTER_INPUT	Filter definition of the input channel Will be adopted once during the initialisation of the function block.	→ <b>FILTER_INPUT (ENUM)</b> (→ p. <a href="#">284</a> )	
uiDetectionTime	UINT	Time delay in [ms] until an error issued after the user-defined upper/lower limit has been exceeded or not reached. Will be adopted once during the initialisation of the function block.  Note: A change in the default settings through the IEC application has an impact on the time response of the diagnostics / function and must be considered by the project engineer when the safety function is set up!	10	Initial value
uiDiagLimitMin	UINT	Lower limit value to diagnose an error. Indication in [μA], [mV], [‰], depends in the set operating mode. Will be adopted once during the initialisation of the function block. The initial value is set so that the FB enters the parameter error if the input value is not assigned. ► Set the input value according to the requirements of the safety function.	16#FFFF = initial value	
uiDiagLimitMax	UINT	Upper limit value to diagnose an error. Indication in [μA], [mV], [‰], depends in the set operating mode. Will be adopted once during the initialisation of the function block. The initial value is set so that the FB enters the parameter error if the input value is not assigned. ► Set the input value according to the requirements of the safety function.	16#0000 = initial value	

## Output parameters

24114

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_ValueDigital	SAFEBOOL	Digital state	FALSE	Initial value. Input in safe state.
			TRUE	Digital modes: The value of the input is within the permissible value range AND there is a valid TRUE level at the input.  Analogue modes: The value of the input is within the permissible value range.
S_ValueAnalog	SAFEUINT	Analogue state	0	Initial value. Input in safe state.
			> 0	Digital modes: Voltage at the input in mV  Analogue modes: Value of the input is within the permissible value range, the value is in the unit of the mode.
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested

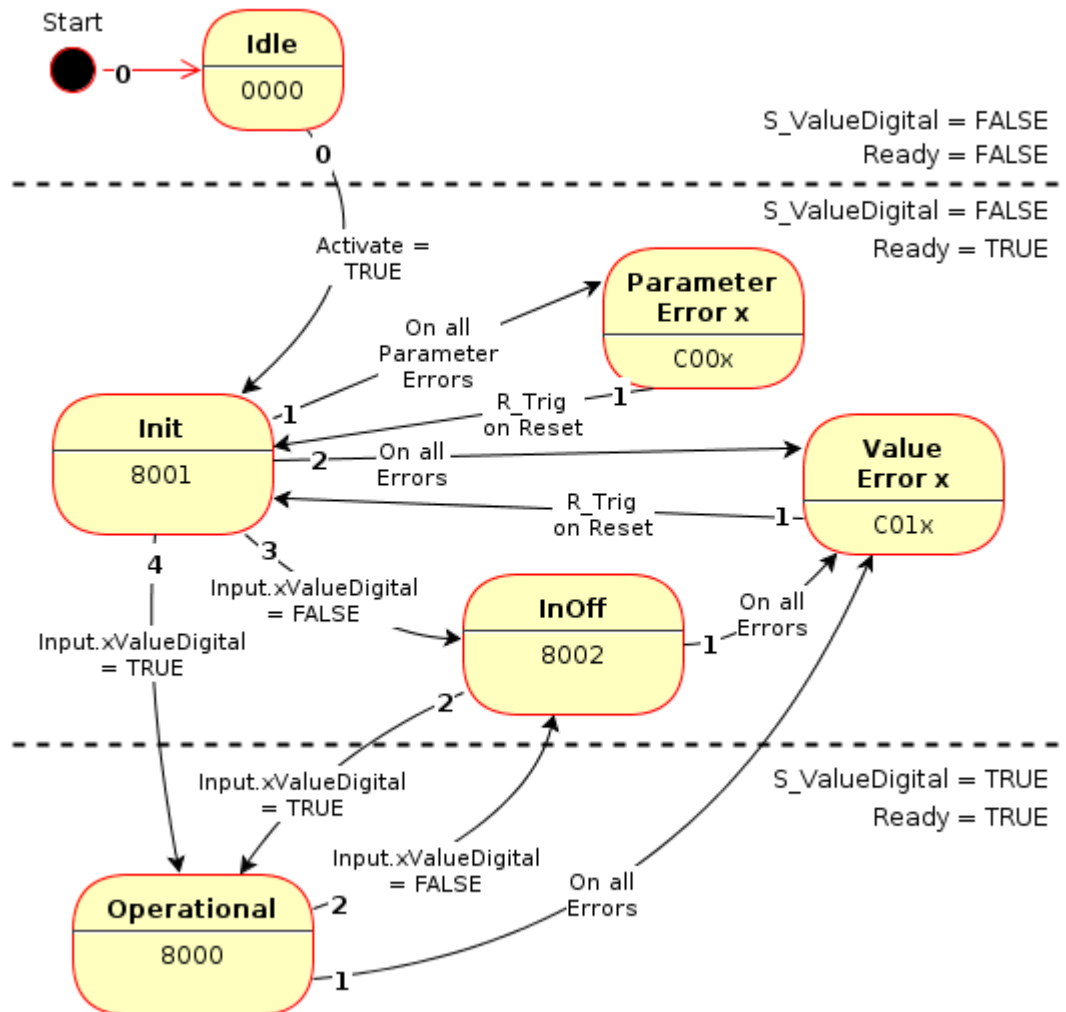


## Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	The function block is active and there is no error state. S_ValueDigital = TRUE S_ValueAnalog > 0 and within the valid value range.
8001	The function block is active, S_ValueDigital = FALSE. Initialisation.
8002	Safety request at the safety input (only digital). S_ValueDigital = FALSE
C001	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiChannel, eMode, eFilter
C002	Error: The selected input channel has not been assigned to the safe PLC.
C003	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiDetectionTime, uiDiagLimitMin, uiDiagLimitMax
C011	System error: one of several system functions cannot be executed
C012	Error: Excessive current at the input. (Only for eMode = IN_CURRENT_CSI)
C013	Error: Analogue input value > uiDiagLimitMax for longer than the time set at uiDetectionTime.
C014	Error: Analogue input value < uiDiagLimitMin for longer than the time set at uiDetectionTime.
C015	Error: Overvoltage at the input.
C016	Error: Undervoltage at the input. (only for eMode = - IN_DIGITAL_CSI - IN_DIGITAL_CSI_NAMUR - IN_VOLTAGE_RATIO)
C017	Error: Input is not calibrated and perhaps inaccurate.
C018	Error: Monitoring diagnostic of the input has detected a deviation.
C019	Error: Monitoring diagnostic of the system voltage has detected a deviation.

## State chart

The status diagram is valid for operation as digital input:



S\_ValueAnalog: A valid analogue value will only be displayed in the states Operational and InOff. In all other states, the value is = 0 (safe state).

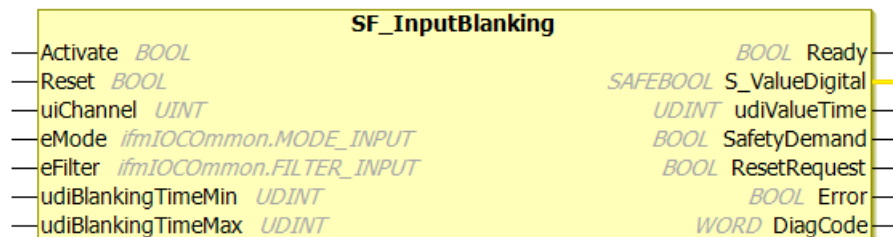
## SF\_InputBlanking

27335

**Function block type:** Safety function block (SF)

**Library:** ifmIOSafety.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### Description

60559

The function block SF\_InputBlanking is used to read safe one-channel inputs receiving blanking signals.



The FB can only be operated at a frequency input with input type IN FREQUENCY-B.

If error messages occur if Activate=FALSE at the assigned input channel, the messages will be indicated directly if Activate=TRUE. The errors must be handled through the application.

The eFilter of the block must be set so that the blanking pulse will not be detected as FALSE. As a reference values, the filter should be set as follows:  $t_{\text{Blanking signal pulse width}} \leq 2 * t_{\text{max(digital)}}$

→ **Filter times of the inputs** (→ p. [477](#))

60559

The function block's behaviour is based on the PLCopen safe function blocks according to the PLCopen specification TC5 Safety part 1. The function block features the extended outputs according to the PLCopen specification TC5 Safety part 3, chapter 2.4.

## Input parameters

27242

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error
uiChannel	UINT	Input channel Will be adopted once during the initialisation of the function block.	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_INPUT	Operating mode of the input channel Will be adopted once during the initialisation of the function block.	→ <b>MODE_INPUT (ENUM)</b> (→ p. 286) Only: IN_DIGITAL_CSI_BLANKING	
eFilter	FILTER_INPUT	Filter definition of the input channel Will be adopted once during the initialisation of the function block.	→ <b>FILTER_INPUT (ENUM)</b> (→ p. 284)	
udiBlankingTimeMin	UDINT	Minimum possible time between 2 blanking pulses in [ms] Will be adopted once during the initialisation of the function block. The initial value is set so that the FB enters the parameter error if the input value is not assigned. ► Set the input value according to the requirements of the safety function.	Permissible value = 0! ► Set value to 0.	
udiBlankingTimeMax	UDINT	Maximum possible time between 2 blanking pulses in [ms] Will be adopted once during the initialisation of the function block. The initial value is set so that the FB enters the parameter error if the input value is not assigned. ► Set the input value according to the requirements of the safety function.	0...4 294 967 295	

## Output parameters

24118

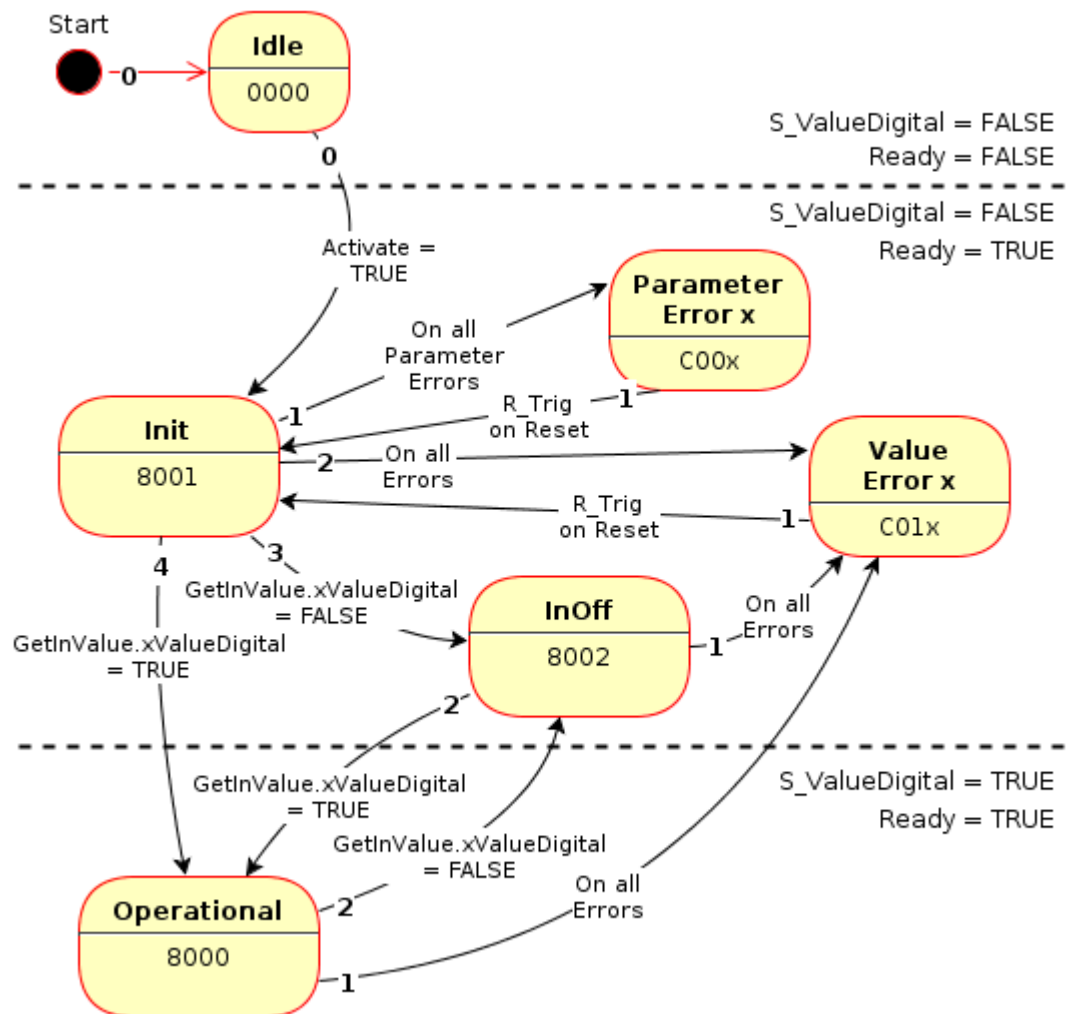
Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_ValueDigital	SAFEBOOL	Digital state	FALSE	initial value. Input in safe state.
			TRUE	Digital modes: The value of the input is TRUE, and blanking pulses have been detected in the valid time frame.
udiValueTime	UDINT	Elapsed time since the last blanking pulse was detected in [µs]	0...4 294 967 295	
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested

Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	The function block is active and there is no error state. Safety output = TRUE
8001	The function block is active, safety output = FALSE. Initialisation.
8002	Safety request at the safety input (only digital). S_ValueDigital = FALSE
C001	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiChannel, eMode, eFilter
C002	Error: The selected input channel has not been assigned to the safe PLC.
C011	System error: one of several system functions cannot be executed
C012	Error: Timeout between two blanking pulses.
C013	Error: Time limit not reached between two blanking pulses.
C014	Error: Overvoltage at the input.
C015	Error: Undervoltage at the input.
C016	Error: Input is not calibrated and perhaps inaccurate.
C017	Error: Monitoring diagnostic of the system voltage has detected a deviation.

## State chart

27357



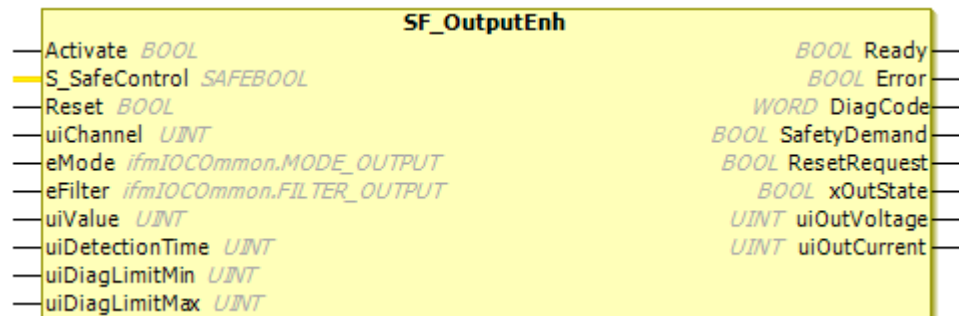
## SF\_OutputEnh

60561

**Function block type:** Safety function block (SF)

**Library:** ifmIOSafety.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

60562

The SF\_OutputEnh block combines the functionalities of the SF\_Output and ConfigDiagLevel blocks:

1. The block is used to control an output that is to be used as a safe output within a safety application.

In addition to the function of the standard FB Output, the block has a safe shutdown of the output channel set at uiChannel by the S\_SafeControl input.

2. The block is used for initial setting of the system-internal error detection of the corresponding output (time delay, upper limit value, lower limit value).



## WARNING

Faulty configuration of the following parameters:

- uiDetectionTime
- uiDiagLimMin / uiDiagLimitMin
- uiDiagLimMax / uiDiagLimitMax


When setting a time behaviour configuration of the lower / upper limit value of input channels/output channels that is different from the factory setting.

- > Risk of personal injuries and/or damage to property.
- > Failure or delay of the safety function is possible.
- ▶ Set upper / lower limit value and check with function test.
- ▶ The uiDetectionTime has an impact on the safety time. The fault will not be detected before the uiDetectionTime has expired.
- ▶ Calculating the safety time.
- ▶ Ensure that the safety time is sufficient to comply with the process safety time ( → **The process safety time** (→ p. [30](#)))



## Input parameters


60563

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_SafeControl	SAFEBOOL	Safe deactivation of the output channel set at uiChannel.	FALSE	Initial value The output channel is in the safe state.
			TRUE	The output channel can be controlled via the uiValue.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error
uiChannel	UINT	Output channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_OUTPUT	Operating mode of the output channel. Permitted mode: OUT_DIGITAL_CSO	→ <b>MODE_OUTPUT (ENUM)</b> (→ p. 287)	
eFilter	FILTER_OUTPUT	Filter definition of the output channel	→ <b>FILTER_OUTPUT (ENUM)</b> (→ p. 285)	
uiValue	UINT	Value that is to be written to the output		
		Possible values	0	Output deactivated
			1	Output activated
uiDetectionTime	UINT	Time delay in [ms] until an error issued after the user-defined upper/lower limit has been exceeded or not reached. Will be adopted once during the initialisation of the function block.  Note: A change in the default settings through the IEC application has an impact on the time response of the diagnostics / function and must be considered by the project engineer when the safety function is set up!	10	Initial value
uiDiagLimitMin	UINT	Lower current limit value to diagnose an error. Specified in [mA]. Will be adopted once during the initialisation of the function block. The initial value is set so that the FB enters the parameter error if the input value is not assigned. ► Set the input value according to the requirements of the safety function.	16#FFFF = initial value	

uiDiagLimitMax	UINT	<p>Upper current limit value to diagnose an error.</p> <p>Specified in [mA].</p> <p>Will be adopted once during the initialisation of the function block.</p> <p>The initial value is set so that the FB enters the parameter error if the input value is not assigned.</p> <p>► Set the input value according to the requirements of the safety function.</p>	16#0000 = initial value
----------------	------	--	-------------------------

## Output parameters

60569

Parameter	Data type	Meaning	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
xOutState	BOOL	Return value activation state of the selected output  The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output is deactivated
			TRUE	Output is activated
uiOutVoltage	UINT	Current output voltage (value in mV) Only available for the operating modes "analogue" and "sensor"	0	Operating mode neither "analogue" nor "sensor"
			≠ 0	Output voltage
uiOutCurrent	UINT	Present output current (value in mA) Not available in the operating modes: <ul style="list-style-type: none"> <li>OUT_DIGITAL_CSI</li> <li>OUT_ANALOGUE_10</li> <li>OUT_SENSOR_05</li> <li>OUT_SENSOR_10</li> </ul>	0...final value of the measuring range	

## Error codes

60570

Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	The function block is active and there is no error state. Safety output = TRUE
8001	Activation = TRUE, the FB is active. Verification whether a manual reset is required.
8002	Function block is ready: Activation = TRUE Function block waits for safety-related input=TRUE
C001	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiChannel, eMode, eFilter
C002	Error: The selected channel has not been assigned to the safe PLC.
C003	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiDetectionTime, uiDiagLimitMin, uiDiagLimitMax
C011	System error: one of several system functions cannot be executed
C012	Error: Stuck-At-High at the output for longer than the set uiDetectionTime.
C013	Error: Stuck-At-Low at the output for longer than the set uiDetectionTime.
C014	Error: Output current > uiDiagLimitMax for longer than the set uiDetectionTime.
C015	Error: Output current < uiDiagLimitMin for longer than the set uiDetectionTime.
C017	Error: Undervoltage of the group supply voltage. No voltage after group switch.
C01B	Error: The output group / output is not calibrated and may be inaccurate.

## SF\_OutGroupEnh

60579

**Function block type:** Safety function block (SF)

**Library:** ifmIOSafety.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

60580

The SF\_OutGroupEnh block combines the functionalities of the SF\_OutGroup and ConfigDiagLevel blocks:

1. The function block controls the activation status of an output group and provides diagnostic information about the group and the connected outputs. The FB can be used to switch an output group including the associated outputs on or off.

In addition, the module has a safe shutdown of the output group by means of the S\_SafeControlinput.

2. The block is used to set the system-internal error detection of the corresponding output (time delay, upper limit value, lower limit value).



### WARNING

Faulty configuration of the following parameters:


- uiDetectionTime
- uiDiagLimMin / uiDiagLimitMin
- uiDiagLimMax / uiDiagLimitMax

When setting a time behaviour configuration of the lower / upper limit value of input channels/output channels that is different from the factory setting.

- > Risk of personal injuries and/or damage to property.
- > Failure or delay of the safety function is possible.
- ▶ Set upper / lower limit value and check with function test.
- ▶ The uiDetectionTime has an impact on the safety time. The fault will not be detected before the uiDetectionTime has expired.
- ▶ Calculating the safety time.
- ▶ Ensure that the safety time is sufficient to comply with the process safety time (→ **The process safety time** (→ p. [30](#)))

## Input parameters

60581

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_SafeControl	SAFEBOOL	Safe shutdown of the output group set at uiChannel including the associated outputs.	FALSE	Initial value The entire output group (OutputGroup) is in the safe state.
			TRUE	The output group (OutputGroup) can be controlled via the xValue input.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error
uiChannel	UINT	Output channel group	→ Data sheet → <b>Note on wiring</b> (→ p. 39)	
eMode	MODE_OUTPUT_GROUP	Operating type of the output channel group	→ <b>MODE_OUTPUT_GROUP (ENUM)</b> (→ p. 302)	
eFilter	FILTER_OUTPUT_GROUP	Defines the limit frequency of the output filter	→ <b>FILTER_OUTPUT_GROUP (ENUM)</b> (→ p. 302)	
xValue	BOOL	Activation requirement for the output group	FALSE	Deactivate output group
			TRUE	Activate output group
uiDetectionTime	UINT	Time delay in [ms] until an error issued after the user-defined upper/lower limit has been exceeded or not reached. Will be adopted once during the initialisation of the function block.  Note: A change in the default settings through the IEC application has an impact on the time response of the diagnostics / function and must be considered by the project engineer when the safety function is set up!	10	Initial value
uiDiagLimitMin	UINT	Lower current limit value to diagnose an error. Specified in [mA]. Will be adopted once during the initialisation of the function block. The initial value is set so that the FB enters the parameter error if the input value is not assigned. ► Set the input value according to the requirements of the safety function.	16#FFFF = initial value	

uiDiagLimitMax	UINT	<p>Upper current limit value to diagnose an error.</p> <p>Specified in [mA].</p> <p>Will be adopted once during the initialisation of the function block.</p> <p>The initial value is set so that the FB enters the parameter error if the input value is not assigned.</p> <p>► Set the input value according to the requirements of the safety function.</p>	16#0000 = initial value
----------------	------	--	-------------------------


60581



The error of an output group will only be reset if all corresponding outputs are error-free.

## Output parameters

60582

Parameter	Data type	Meaning	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
xGroupState	BOOL	Return value activation state of the selected output group  The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output group is deactivated
			TRUE	Output value is activated
uiGroupCurrent	UINT	Measured output current of the entire group in [mA]	available = 0...final value of the measuring range	
uiVBBxVoltage	UINT	Measured voltage before the group switch in [mV]	available = 0...final value of the measuring range	
uiGroupVoltage	UINT	Measured voltage after the group switch in [mV]	available = 0...final value of the measuring range	

## Error codes

60583

Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	The function block is active and there is no error state. Safety output = TRUE
8001	Activation = TRUE, the FB is active. Verification whether a manual reset is required.
8002	Function block is ready: Activation = TRUE Function block waits for safety-related input=TRUE

Value [hex]	Description
C001	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiChannel, eMode, eFilter
C002	Error: The selected channel has not been assigned to the safe PLC.
C003	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiDetectionTime, uiDiagLimitMin, uiDiagLimitMax
C011	System error: one of several system functions cannot be executed
C012	Error: Stuck-At-High at the output for longer than the set uiDetectionTime.
C013	Error: Stuck-At-Low at the output for longer than the set uiDetectionTime.
C014	Error: Output current > uiDiagLimitMax for longer than the set uiDetectionTime.
C015	Error: Output current < uiDiagLimitMin for longer than the set uiDetectionTime.
C016	Error: Overvoltage of the group supply voltage VBBx.
C017	Error: Undervoltage of group supply voltage VBBx.
C018	Error: Start-up test of the group switch has failed. Possible causes: VBBx is not supplied. At least one output of the group is supplied externally. At least one output detects STUCK_AT_HIGH. Defect at group switch or output
C019	Error: At least one output of the output group is in the error state.
C01B	Error: The output group / output is not calibrated and may be inaccurate.



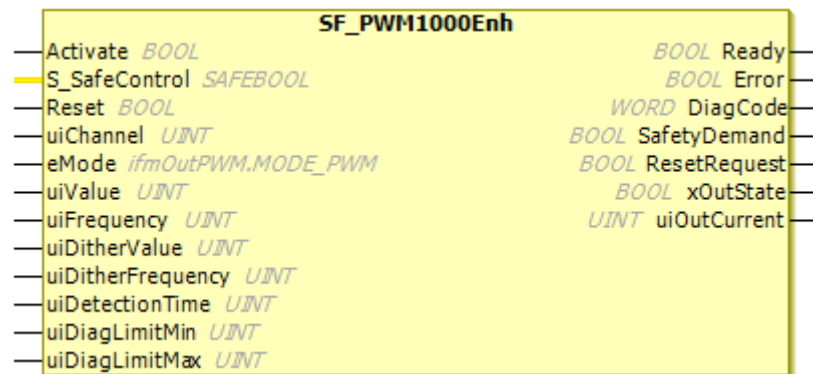
## SF\_PWM1000Enh

60589

**Function block type:** Safety function block (SF)

**Library:** ifmIOSafety.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

60590

The block SF\_PWM1000Enh combines the functionalities of the blocks SF\_PWM1000 and ConfigDiagLevel:

1. The function block is used to configure and to operate an output with pulse width modulation.

In addition to the function of the standard function block, the latter features safe deactivation of the uiOutCurrent output by the S\_SafeControl input.

2. The block is used to set the system-internal error detection of the corresponding output (time delay, upper limit value, lower limit value).



## WARNING

Faulty configuration of the following parameters:


- uiDetectionTime
- uiDiagLimMin / uiDiagLimitMin
- uiDiagLimMax / uiDiagLimitMax

When setting a time behaviour configuration of the lower / upper limit value of input channels/output channels that is different from the factory setting.

- > Risk of personal injuries and/or damage to property.
- > Failure or delay of the safety function is possible.
- ▶ Set upper / lower limit value and check with function test.
- ▶ The uiDetectionTime has an impact on the safety time. The fault will not be detected before the uiDetectionTime has expired.
- ▶ Calculating the safety time.
- ▶ Ensure that the safety time is sufficient to comply with the process safety time ( → **The process safety time** (→ p. [30](#)))

## Input parameters


60591

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_SafeControl	SAFEBOOL	Safe deactivation of the output channel set at uiChannel.	FALSE	Initial value The output channel is in the safe state.
			TRUE	The output channel can be controlled via the uiValue.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error
uiChannel	UINT	Output channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_OUTPUT	Operating type of the output channel	→ <b>MODE_OUTPUT (ENUM)</b> (→ p. 287)	
uiValue	UNIT	Pulse/pause ration of the PWM output signal in [%]	permissible = 0...1000	
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet	
uiDitherValue	UNIT	Peak-to-peak value of the dither signal which overlays with the PWM signal, in [%]	permissible = 0...1000 If the resulting PWM ratio value is outside the 0...1000 % range, the dither value will be temporarily internally reduced to the minimum/maximum value that is possible, so that the mean value of the PWM ratio corresponds with the required value.	
uiDitherFrequency	UNIT	Frequency for the dither signal at the PWM output in [Hz]	permissible = $0 \dots \text{uiFrequency} / 2$ The value at uiDitherFrequency must be an integer part of the value indicated to uiFrequency. Examples: uiFrequency = 300 Hz uiDitherFrequency = 50 Hz $\Rightarrow 300 / 50 = 6$ $\Rightarrow$ even factor, valid uiDitherFrequency = 100 Hz $\Rightarrow 300 / 100 = 3$ $\Rightarrow$ uneven factor, invalid	
uiDetectionTime	UINT	Time delay in [ms] until an error issued after the user-defined upper/lower limit has been exceeded or not reached. Will be adopted once during the initialisation of the function block.  Note: A change in the default settings through the IEC application has an impact on the time response of the diagnostics / function and must be considered by the project engineer when the safety function is set up!	10	Initial value

uiDiagLimitMin	UINT	<p>Lower current limit value to diagnose an error.</p> <p>Specified in [mA].</p> <p>Will be adopted once during the initialisation of the function block.</p> <p>The initial value is set so that the FB enters the parameter error if the input value is not assigned.</p> <p>► Set the input value according to the requirements of the safety function.</p>	16#FFFF = initial value
uiDiagLimitMax	UINT	<p>Upper current limit value to diagnose an error.</p> <p>Specified in [mA].</p> <p>Will be adopted once during the initialisation of the function block.</p> <p>The initial value is set so that the FB enters the parameter error if the input value is not assigned.</p> <p>► Set the input value according to the requirements of the safety function.</p>	16#0000 = initial value

## Output parameters

60592

Parameter	Data type	Meaning	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
xOutState	BOOL	Return value activation state of the selected output  The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output is deactivated
			TRUE	Output is activated
uiOutCurrent	UINT	Present output current in [mA]	available = 0...final value of the measuring range	

## Error codes

60593

Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	The function block is active and there is no error state. Safety output = TRUE
8001	Activation = TRUE, the FB is active. Verification whether a manual reset is required.
8002	Function block is ready: Activation = TRUE Function block waits for safety-related input=TRUE
C001	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiChannel, eMode, eFilter
C002	Error: The selected channel has not been assigned to the safe PLC.

Value [hex]	Description
C003	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiDetectionTime, uiDiagLimitMin, uiDiagLimitMax
C004	Error: Wrong parameter at uiFrequency.
C005	Error: Wrong parameter at uiDitherFrequency or uiDitherValue.
C011	System error: one of several system functions cannot be executed
C012	Error: Stuck-At-High at the output for longer than the set uiDetectionTime.
C013	Error: Stuck-At-Low at the output for longer than the set uiDetectionTime.
C014	Error: Output current > uiDiagLimitMax for longer than the set uiDetectionTime.
C015	Error: Output current < uiDiagLimitMin for longer than the set uiDetectionTime.
C017	Error: Undervoltage of the group supply voltage. No voltage after group switch.
C01B	Error: The output group / output is not calibrated and may be inaccurate.

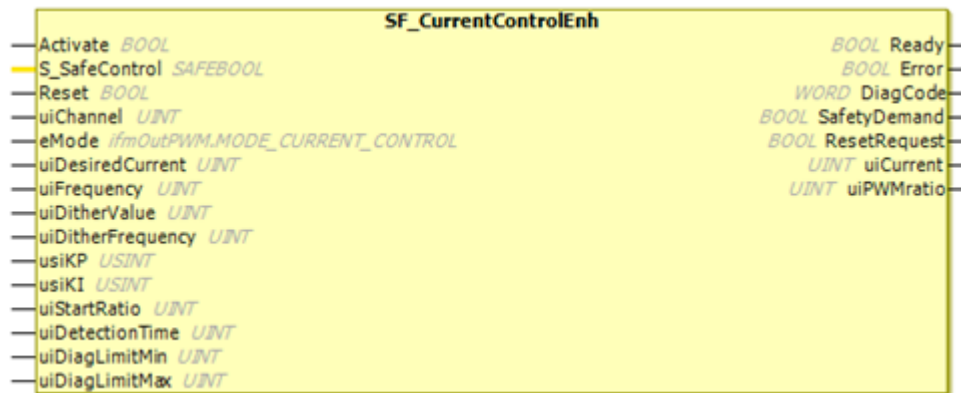
## SF\_CurrentControlEnh

60597

**Function block type:** Safety function block (SF)

**Library:** ifmIOSafety.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

60598

The SF\_CurrentControlEnh block combines the functionalities of the blocks SF\_CurrentControl and ConfigDiagLevel:

1. The function block is used to configure and operate a current controlled output.

The current control is provided by means of pulse width modulation (PWM). The configuration of PWM frequency and dither is also done with this FB.

In addition to the function of the standard function block, the SF features safe deactivation of the output channel set at uiChannel by the S\_SafeControl input.

2. The block is used to set the system-internal error detection of the corresponding output (time delay, upper limit value, lower limit value).



## WARNING

Faulty configuration of the following parameters:

- uiDetectionTime
- uiDiagLimMin / uiDiagLimitMin
- uiDiagLimMax / uiDiagLimitMax


When setting a time behaviour configuration of the lower / upper limit value of input channels/output channels that is different from the factory setting.

- > Risk of personal injuries and/or damage to property.
- > Failure or delay of the safety function is possible.
- ▶ Set upper / lower limit value and check with function test.
- ▶ The uiDetectionTime has an impact on the safety time. The fault will not be detected before the uiDetectionTime has expired.
- ▶ Calculating the safety time.
- ▶ Ensure that the safety time is sufficient to comply with the process safety time ( → **The process safety time** (→ p. 30))

## Input parameters

24129

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_SafeControl	SAFEBOOL	Safe deactivation of the output channel set at uiChannel.	FALSE	Initial value The output channel is in the safe state.
			TRUE	The output channel can be controlled via the uiDesiredCurrent.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error
uiChannel	UINT	Output channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_CURRENT_CONTROL	Operating type of the output channel	→ <b>MODE_CURRENT_CONTROL (ENUM)</b> (→ p. 316)	
uiDesiredCurrent	UINT	Default value at the output channel. When 0 is set, the output is immediately deactivated.	0 ... 65535	
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet	

uiDitherValue	UNIT	Peak-to-peak value of the dither signal which overlays with the PWM signal, in [%]	permissible = 0...1000 If the resulting PWM ratio value is outside the 0...1000 % range, the dither value will be temporarily internally reduced to the minimum/maximum value that is possible, so that the mean value of the PWM ratio corresponds with the required value.	
uiDitherFrequency	UNIT	Frequency for the dither signal at the PWM output in [Hz]	permissible = 0...uiFrequency / 2 The value at uiDitherFrequency must be an integer part of the value indicated to uiFrequency. Examples: uiFrequency = 300 Hz uiDitherFrequency = 50 Hz $\Rightarrow$ 300 / 50 = 6 $\Rightarrow$ even factor, valid uiDitherFrequency = 100 Hz $\Rightarrow$ 300 / 100 = 3 $\Rightarrow$ uneven factor, invalid	
usiKP	USINT	Proportional component of the output signal	0...255 If usiKP = 0 then no control.	
usiKI	USINT	Integral component of the output signal	0 ... 255 If usiKI = 0 then no control.	
uiStartRatio	UNIT	Mark-to-space ratio of the PWM output signal after start of the controller in [%]	permissible = 0...1000 Recommended standard start value: 500	
uiDetectionTime	UINT	Time delay in [ms] until an error issued after the user-defined upper/lower limit has been exceeded or not reached. Will be adopted once during the initialisation of the function block.  Note: A change in the default settings through the IEC application has an impact on the time response of the diagnostics / function and must be considered by the project engineer when the safety function is set up!	10	Initial value
uiDiagLimitMin	UINT	Lower current limit value to diagnose an error. Specified in [mA]. Will be adopted once during the initialisation of the function block. The initial value is set so that the FB enters the parameter error if the input value is not assigned. ► Set the input value according to the requirements of the safety function.	16#FFFF = initial value	
uiDiagLimitMax	UINT	Upper current limit value to diagnose an error. Specified in [mA]. Will be adopted once during the initialisation of the function block. The initial value is set so that the FB enters the parameter error if the input value is not assigned. ► Set the input value according to the requirements of the safety function.	16#0000 = initial value	



## Output parameters

60600

Parameter	Data type	Meaning	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
uiCurrent	UINT	Output current signal in [mA]	available = 0...final value of the measuring range	
uiPWMRatio	UINT	PWM pulse ration calculated by the PI controller in [%]		

## Error codes

60593

Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	The function block is active and there is no error state. Safety output = TRUE
8001	Activation = TRUE, the FB is active. Verification whether a manual reset is required.
8002	Function block is ready: Activation = TRUE Function block waits for safety-related input=TRUE
C001	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiChannel, eMode, eFilter
C002	Error: The selected channel has not been assigned to the safe PLC.
C003	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiDetectionTime, uiDiagLimitMin, uiDiagLimitMax
C004	Error: Wrong parameter at uiFrequency.
C005	Error: Wrong parameter at uiDitherFrequency or uiDitherValue.

Value [hex]	Description
C011	System error: one of several system functions cannot be executed
C012	Error: Stuck-At-High at the output for longer than the set uiDetectionTime.
C013	Error: Stuck-At-Low at the output for longer than the set uiDetectionTime.
C014	Error: Output current > uiDiagLimitMax for longer than the set uiDetectionTime.
C015	Error: Output current < uiDiagLimitMin for longer than the set uiDetectionTime.
C017	Error: Undervoltage of the group supply voltage. No voltage after group switch.
C01B	Error: The output group / output is not calibrated and may be inaccurate.

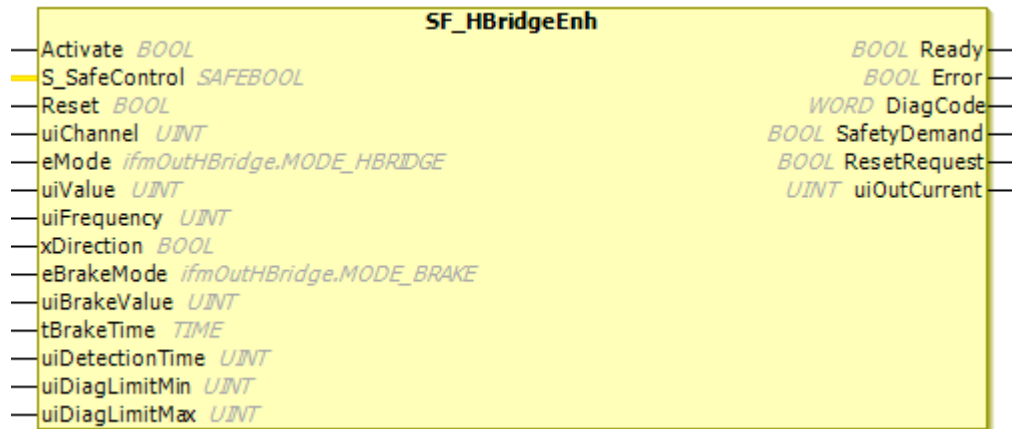
## SF\_HBridgeEnh

60602

**Function** Safety function block (SF)  
**block type:**

**Library:** ifmIOSafety.library

**Symbol in  
CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

60603

The SF\_HBridgeEnh block combines the functionalities of the blocks SF\_HBridge and ConfigDiagLevel:

1. The block configures and controls a pair of output channels in the "HBridge" operating type to control a motor.

In addition to the function of the standard FB HBridge, it has a safe deactivation of the output channel set at uiChannel by the input S\_SafeControl.

When the block is called up for the first time, both outputs (channel A and channel B) will be tested. The test may take several cycles before the H-bridge can be used.

2. The block is used to set the system-internal error detection of the corresponding output (time delay, upper limit value, lower limit value).



## WARNING

Faulty configuration of the following parameters:

- uiDetectionTime
- uiDiagLimMin / uiDiagLimitMin
- uiDiagLimMax / uiDiagLimitMax


When setting a time behaviour configuration of the lower / upper limit value of input channels/output channels that is different from the factory setting.

- > Risk of personal injuries and/or damage to property.
- > Failure or delay of the safety function is possible.
- ▶ Set upper / lower limit value and check with function test.
- ▶ The uiDetectionTime has an impact on the safety time. The fault will not be detected before the uiDetectionTime has expired.
- ▶ Calculating the safety time.
- ▶ Ensure that the safety time is sufficient to comply with the process safety time ( → **The process safety time** (→ p. 30))

## Input parameters

60604

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_SafeControl	SAFEBOOL	Safe deactivation of the output channel set at uiChannel.	FALSE	Initial value The output channel is in the safe state.
			TRUE	The output channel can be controlled via the uiValue.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error
uiChannel	UINT	1. Output channel (channel A) of the output channel pair	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Example:	
			106	Group 1 + channel 6 + 7
eMode	MODE_HBRIDGE	Operating mode	→ <b>MODE_HBRIDGE (ENUM)</b> (→ p. 307)	
uiValue	UNIT	Pulse/pause ration of the PWM output signal in [%]	permissible = 0...1000	
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet	
xDirection	BOOL	The direction in which the current flows via the bridge connections. Determines the direction of rotation of the connected motor.	FALSE	PWM Current Sourcing (CSO) is on channel A
			TRUE	PWM Current Sourcing (CSO) is on channel B
eBrakeMode	MODE_BRAKE	Brake mode that applies when the direction of rotation is changed or when stopping	→ <b>MODE_BRAKE (ENUM)</b> (→ p. 307)	

uiBrakeValue	UINT	Pulse/pause ratio of the PWM output signal at the corresponding current sinking output of the bridge in [%] The input is only relevant in the eBrakeModes that end with "_DYNAMIC" (= dynamic brake).	permissible = 0...1 000	
tBrakeTime	TIME	Indicates the braking time for the current sinking side of the bridge The input is only relevant in eBrakeModes ending with "_BTIME".	permissible = 0...1 h	
uiDetectionTime	UINT	Time delay in [ms] until an error issued after the user-defined upper/lower limit has been exceeded or not reached. Will be adopted once during the initialisation of the function block.  Note: A change in the default settings through the IEC application has an impact on the time response of the diagnostics / function and must be considered by the project engineer when the safety function is set up!	10	Initial value
uiDiagLimitMin	UINT	Lower current limit value to diagnose an error. Specified in [mA]. Will be adopted once during the initialisation of the function block. The initial value is set so that the FB enters the parameter error if the input value is not assigned. ► Set the input value according to the requirements of the safety function.	16#FFFF = initial value	
uiDiagLimitMax	UINT	Upper current limit value to diagnose an error. Specified in [mA]. Will be adopted once during the initialisation of the function block. The initial value is set so that the FB enters the parameter error if the input value is not assigned. ► Set the input value according to the requirements of the safety function.	16#0000 = initial value	

## Output parameters

60605

Parameter	Data type	Meaning	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

Parameter	Data type	Meaning	Possible values	
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
uiOutCurrent	UINT	Measured current at the PWM output during normal operation in [mA] When braking, uiOutCurrent is = 0 because no regular current exists in the lowside path.	available = 0...final value of the measuring range	

## Error codes

60606

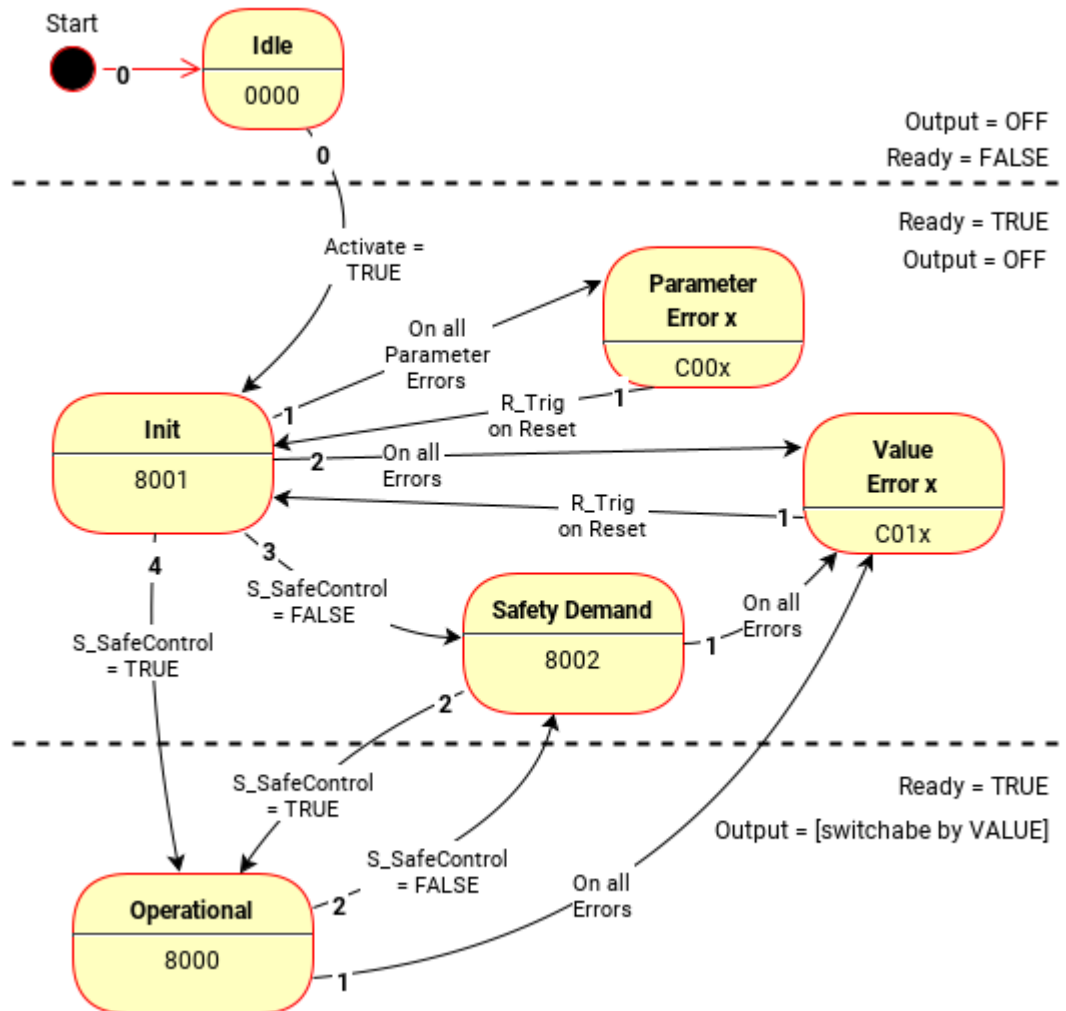
Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	The function block is active and there is no error state. Safety output = TRUE
8001	Activation = TRUE, the FB is active. Verification whether a manual reset is required.
8002	Function block is ready: Activation = TRUE Function block waits for safety-related input=TRUE
C001	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiChannel, eMode, eFilter
C002	Error: The selected channel has not been assigned to the safe PLC.
C003	Error: On at least one of the following function bloc inputs, a wrong parameter has been set: uiDetectionTime, uiDiagLimitMin, uiDiagLimitMax
C004	Error: Wrong parameter at uiFrequency.
C011	System error: one of several system functions cannot be executed
C012	Error: Stuck-At-High at the output for longer than the set uiDetectionTime.
C013	Error: Stuck-At-Low at the output for longer than the set uiDetectionTime.
C014	Error: Output current > uiDiagLimitMax for longer than the set uiDetectionTime.
C015	Error: Output current < uiDiagLimitMin for longer than the set uiDetectionTime.
C017	Error: Undervoltage of the group supply voltage. No voltage after group switch.
C01A	Error: Startup test of the H-Bridge has failed.
C01B	Error: The output group / output is not calibrated and may be inaccurate.

## State diagram SF\_[Type]Enh

60608

Status diagram for the safety function blocks SF\_OutputEnh, SF\_OutGroupEnh, SF\_PWM1000Enh, SF\_CurrentControlEnh, SF\_HBridgeEnh:



In addition to the diagram, the following conditions apply:

- The transition from any state into the Idle state through `Activate = FALSE` is not shown here for the sake of clarity. This transition always has priority 0 = highest priority.
- If the input `Reset=TRUE` and there is no other error, the function block changes from the Value Error x to the Operational state.
- If there is a Value Error x, and an error of higher priority occurs, the function block changes to the higher error state Value Error x.
- If the input `Reset =TRUE` and one or several other errors are active, the function block changes to this Value Error x according to the priority.
- Lower error priority = higher priority value, high error priority= low priority value



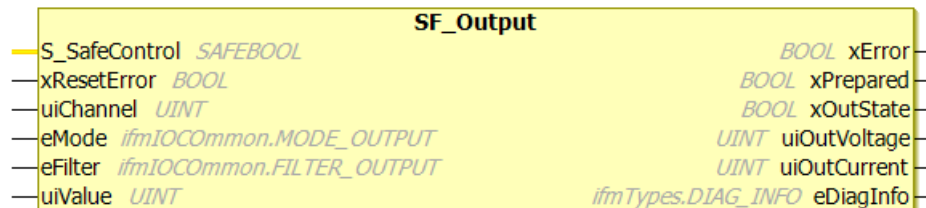
## SF\_Output

27339

**Function block type:** Safety function block (SF)

**Library:** ifmIOSafety.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))



This function block can continue to be used in existing applications for reasons of compatibility.

- Recommendation: In new applications, use the enhanced variant of this function block (SF\_[Type]\_Enh).

## Description

24120

The function block SF\_Output is used to control an output that is to be used as a safe output within a safety application.

In addition to the function of the standard function block Output, the block features safe deactivation of the output channel set at uiChannel by the S\_SafeControl input.

## Input parameters


27243

Parameter	Data type	Description	Possible values	
S_SafeControl	SAFEBOOL	Safe deactivation of the output channel set at uiChannel.	FALSE	Initial value The output channel is in the safe state.
			TRUE	The output channel can be controlled via the uiValue.
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	Output channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. <a href="#">39</a> )	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_OUTPUT	Operating type of the output channel	→ <b>MODE_OUTPUT (ENUM)</b> (→ p. <a href="#">287</a> )	
eFilter	FILTER_OUTPUT	Filter definition of the output channel	→ <b>FILTER_OUTPUT (ENUM)</b> (→ p. <a href="#">285</a> )	

uiValue	UINT	Value that is to be written to the output		
		In the digital mode or sensor supply mode; if setting at the eMode input =	0	Output deactivated
		<ul style="list-style-type: none"> <li>▪ OUT_DIGITAL_CSI</li> <li>▪ OUT_DIGITAL_CSO</li> <li>▪ OUT_SENSOR_05</li> <li>▪ OUT_SENSOR_10</li> </ul>	1	Output activated
		In analogue mode; if setting at the eMode input =	0...10 000	
		<ul style="list-style-type: none"> <li>▪ OUT_ANALOGUE_10</li> </ul> Values indicated in [mV]		

## Output parameters

27077

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>▪ Error occurred</li> <li>▪ Action could not be executed</li> <li>▪ Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
xOutState	BOOL	Return value activation state of the selected output  The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output is deactivated
			TRUE	Output is activated
uiOutVoltage	UINT	Current output voltage (value in mV) Only available for the operating modes "analogue" and "sensor"	0	Operating mode neither "analogue" nor "sensor"
			≠ 0	Output voltage
uiOutCurrent	UINT	Present output current (value in mA) Not available in the operating modes: <ul style="list-style-type: none"> <li>▪ OUT_DIGITAL_CSI</li> <li>▪ OUT_ANALOGUE_10</li> <li>▪ OUT_SENSOR_05</li> <li>▪ OUT_SENSOR_10</li> </ul>	0...final value of the measuring range	

Diagnostic codes (→**Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING                      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE                            State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE                      State: FB/Function is inactive.
- ERR\_INTERNAL                        Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED                      Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION    Error: Internal communication error
- ERR\_STUCK\_AT\_HIGH                Error: Signal frozen, signal state high.
- ERR\_STUCK\_AT\_LOW                 Error: Signal frozen, signal state low.
- ERR\_OVERLOAD\_CURRENT            Error: Maximum current exceeded.
- ERR\_LOW\_CURRENT                  Error: Minimum current not reached.
- DIAG\_INVALID\_VALUE                At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB                     The selected resource has no valid calibration.  
The displayed values are maybe faulty.
- DIAG\_ACCESS                        FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_STUCK\_AT\_HIGH               Error: Signal frozen, signal state high.
- DIAG\_STUCK\_AT\_LOW                Error: Signal frozen, signal state low.
- DIAG\_OVERLOAD\_CURRENT           Maximum current exceeded.
- DIAG\_LOW\_CURRENT                 Minimum current not reached.
- DIAG\_INVALID\_FILTER               One or several filter settings are invalid.
- DIAG\_UNDERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs: Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.

## SF\_OutGroup

27338

**Function block type:** Safety function block (SF)

**Library:** ifmIOSafety.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))



This function block can continue to be used in existing applications for reasons of compatibility.

- Recommendation: In new applications, use the enhanced variant of this function block (SF\_[Type]\_Enh).

## Description

27146

The function block SF\_OutGroup controls the activation status of an output group and provides diagnostic information about the group and the connected outputs. Using the FB, an output group including the corresponding outputs can be switched on or off.

The SF\_OutGroup additionally features safe deactivation of the output group by the S\_SafeControl input.

## Input parameters

27244

Parameter	Data type	Description	Possible values	
S_SafeControl	SAFEBOOL	Safe shutdown of the output group set at uiChannel including the associated outputs.	FALSE	Initial value The entire output group (OutputGroup) is in the safe state.
			TRUE	The output group (OutputGroup) can be controlled via the xValue input.
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	Output channel group	→ Data sheet → <b>Note on wiring</b> (→ p. <a href="#">39</a> )	
eMode	MODE_ OUTPUT_ GROUP	Operating type of the output channel group	→ <b>MODE_OUTPUT_GROUP (ENUM)</b> (→ p. <a href="#">302</a> )	
eFilter	FILTER_ OUTPUT_ GROUP	Defines the limit frequency of the output filter	→ <b>FILTER_OUTPUT_GROUP (ENUM)</b> (→ p. <a href="#">302</a> )	
xValue	BOOL	Activation requirement for the output group	FALSE	Deactivate output group
			TRUE	Activate output group


27244



The error of an output group will only be reset if all corresponding outputs are error-free.

## Output parameters

23329

Parameter	Data type	Meaning	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
xGroupState	BOOL	Return value activation state of the selected output group  The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output group is deactivated
			TRUE	Output value is activated
uiGroupCurrent	UINT	Measured output current of the entire group in [mA]	available = 0...final value of the measuring range	
uiVBBxVoltage	UINT	Measured voltage before the group switch in [mV]	available = 0...final value of the measuring range	
uiGroupVoltage	UINT	Measured voltage after the group switch in [mV]	available = 0...final value of the measuring range	

Diagnostic codes (→**Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING                      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE                            State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE                      State: FB/Function is inactive.
- ERR\_INTERNAL                        Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED                      Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION    Error: Internal communication error
- ERR\_OVERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage exceeded.
  - For outputs: Error: Voltage of the corresponding output groups supply or at VBB30 / VBB15 not reached.
- ERR\_OVERLOAD\_CURRENT            Error: Maximum current exceeded.
- ERR\_GROUP\_SW\_TEST                Error: Start-up test of the group switch has failed.  
Possible causes:
  - VBBx is not supplied.
  - At least one output of the group is supplied externally.
  - At least one output detects STUCK\_AT\_HIGH.
  - Defect at group switch or output
- ERR\_AT\_GROUP\_OUTPUT             Error: At least one output of the output group is in an error state.
- ERR\_STUCK\_AT\_HIGH                Error: Signal frozen, signal state high.
- ERR\_STUCK\_AT\_LOW                 Error: Signal frozen, signal state low.
- ERR\_UNDERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs: Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG\_INVALID\_VALUE                At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB                      The selected resource has no valid calibration.  
The displayed values are maybe faulty.
- DIAG\_ACCESS                        FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_OVERLOAD\_CURRENT            Maximum current exceeded.
- DIAG\_AT\_GROUP\_OUTPUT             At least one output of the output group is in the error state.
- DIAG\_INVALID\_FILTER                One or several filter settings are invalid.
- DIAG\_STUCK\_AT\_HIGH                Error: Signal frozen, signal state high.
- DIAG\_STUCK\_AT\_LOW                Error: Signal frozen, signal state low.

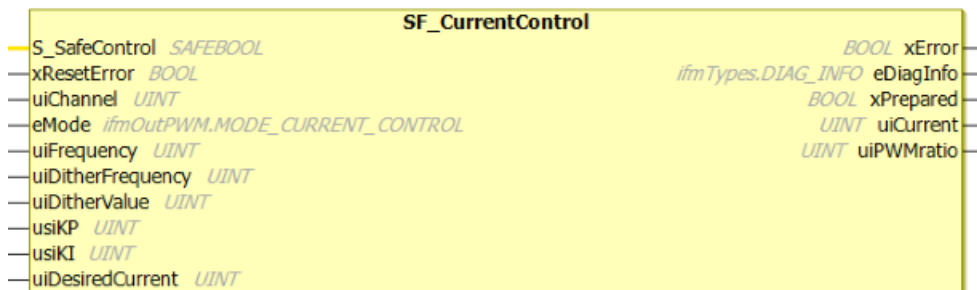
## SF\_CurrentControl

27319

**Function block type:** Safety function block (SF)

**Library:** ifmIOSafety.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))



This function block can continue to be used in existing applications for reasons of compatibility.

- Recommendation: In new applications, use the enhanced variant of this function block (SF\_[Type]\_Enh).

## Description

24128

The function block SF\_CurrentControl is used to configure and operate a current controlled output. The current control is provided by means of pulse width modulation (PWM). The configuration of PWM frequency and dither is also done with this FB.

In addition to the function of the standard function block, the SF features safe deactivation of the output channel set at uiChannel by the S\_SafeControl input.



## Input parameters

Parameter	Data type	Description	Possible values	
S_SafeControl	SAFEBOOL	Safe deactivation of the output channel set at uiChannel.	FALSE	Initial value The output channel is in the safe state.
			TRUE	The output channel can be controlled via the uiDesiredCurrent.
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	Output channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_CURRENT_CONTROL	Operating type of the output channel	→ <b>MODE_CURRENT_CONTROL (ENUM)</b> (→ p. 316)	
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet	
uiDitherFrequency	UNIT	Frequency for the dither signal at the PWM output in [Hz]	permissible = $0 \dots \text{uiFrequency} / 2$ The value at uiDitherFrequency must be an integer part of the value indicated to uiFrequency. Examples: uiFrequency = 300 Hz uiDitherFrequency = 50 Hz ⇒ $300 / 50 = 6$ ⇒ even factor, valid uiDitherFrequency = 100 Hz ⇒ $300 / 100 = 3$ ⇒ uneven factor, invalid	
uiDitherValue	UNIT	Peak-to-peak value of the dither signal which overlays with the PWM signal, in [‰]	permissible = $0 \dots 1000$ If the resulting PWM ratio value is outside the $0 \dots 1000$ ‰ range, the dither value will be temporarily internally reduced to the minimum/maximum value that is possible, so that the mean value of the PWM ratio corresponds with the required value.	
usiKP	USINT	Proportional component of the output signal	0...255 If usiKP = 0 then no control.	
usiKI	USINT	Integral component of the output signal	0 ... 255 If usiKI = 0 then no control.	
uiDesiredCurrent	UINT	Default value at the output channel. When 0 is set, the output is immediately deactivated.	0 ... 65535	

## Output parameters

27086

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
uiCurrent	UINT	Output current signal in [mA]	available = 0...final value of the measuring range	
uiPWMRatio	UINT	PWM pulse ration calculated by the PI controller in [%]		

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. 463)):

- STAT\_PREPARING State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE State: FB/Function is inactive.
- ERR\_INTERNAL Error: Internal system error
  - Contact the ifm Service Center!
- ERR\_UNDEFINED Error: Unknown error
  - Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION Error: Internal communication error
- ERR\_STUCK\_AT\_HIGH Error: Signal frozen, signal state high.
- ERR\_OVERLOAD\_CURRENT Error: Maximum current exceeded.
- ERR\_LOW\_CURRENT Error: Minimum current not reached.
- DIAG\_INVALID\_VALUE At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB The selected resource has no valid calibration. The displayed values are maybe faulty.
- DIAG\_ACCESS FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_STUCK\_AT\_HIGH Error: Signal frozen, signal state high.
- DIAG\_OVERLOAD\_CURRENT Maximum current exceeded.
- DIAG\_LOW\_CURRENT Minimum current not reached.
- DIAG\_UNDERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs: Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG\_INVALID\_FREQUENCY The frequency is not supported in the active mode.
- DIAG\_INVALID\_DITHER At least one dither parameter is invalid.

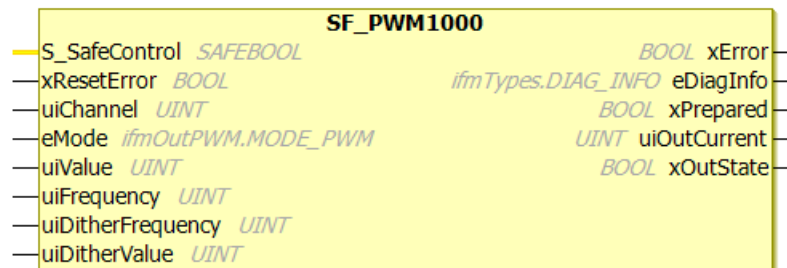
## SF\_PWM1000

27340

**Function block type:** Safety function block (SF)

**Library:** ifmIOSafety.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))



This function block can continue to be used in existing applications for reasons of compatibility.

- Recommendation: In new applications, use the enhanced variant of this function block (SF\_[Type]\_Enh).

## Description

24132

The function block SF\_PWM1000 is used to configure and to operate an output with pulse width modulation.

In addition to the function of the standard function bloc, the latter features safe deactivation of the output channel set at uiChannel by the S\_SafeControl input.




If the function block ConfigDiagLevel is used:

- Please observe the formula to calculate the DetectionTime → **ConfigDiagLevel** (→ p. [290](#))

## Input parameters

Parameter	Data type	Description	Possible values	
S_SafeControl	SAFEBOOL	Safe deactivation of the output channel set at uiChannel.	FALSE	Initial value The output channel is in the safe state.
			TRUE	The output channel can be controlled via the uiValue.
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	Output channel	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. 39)	
			Examples:	
			403	Group 4 + channel 3
			502	Group 5 + channel 2
eMode	MODE_PWM	Operating type of the output channel	→ <b>MODE_PWM (ENUM)</b> (→ p. 316)	
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet	
uiValue	UNIT	Pulse/pause ration of the PWM output signal in [‰]	permissible = 0...1000	
uiDitherFrequency	UNIT	Frequency for the dither signal at the PWM output in [Hz]	permissible = $0 \dots \text{uiFrequency} / 2$ The value at uiDitherFrequency must be an integer part of the value indicated to uiFrequency. Examples: uiFrequency = 300 Hz uiDitherFrequency = 50 Hz ⇒ $300 / 50 = 6$ ⇒ even factor, valid uiDitherFrequency = 100 Hz ⇒ $300 / 100 = 3$ ⇒ uneven factor, invalid	
uiDitherValue	UNIT	Peak-to-peak value of the dither signal which overlays with the PWM signal, in [‰]	permissible = 0...1000 If the resulting PWM ratio value is outside the 0...1000 ‰ range, the dither value will be temporarily internally reduced to the minimum/maximum value that is possible, so that the mean value of the PWM ratio corresponds with the required value.	

## Output parameters

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
uiOutCurrent	UINT	Present output current in [mA]	available = 0...final value of the measuring range	
xOutState	BOOL	Return value activation state of the selected output  The state may deviate from the required output state if e.g. a safety function has deactivated an output group due to an error.	FALSE	Output is deactivated
			TRUE	Output is activated

Diagnostic codes (→**Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING                      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE                            State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE                      State: FB/Function is inactive.
- ERR\_INTERNAL                        Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED                      Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION    Error: Internal communication error
- ERR\_STUCK\_AT\_HIGH                Error: Signal frozen, signal state high.
- ERR\_OVERLOAD\_CURRENT            Error: Maximum current exceeded.
- ERR\_LOW\_CURRENT                  Error: Minimum current not reached.
- DIAG\_INVALID\_VALUE                At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB                      The selected resource has no valid calibration.  
The displayed values are maybe faulty.
- DIAG\_ACCESS                        FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_STUCK\_AT\_HIGH                Error: Signal frozen, signal state high.
- DIAG\_OVERLOAD\_CURRENT            Maximum current exceeded.
- DIAG\_LOW\_CURRENT                  Minimum current not reached.
- DIAG\_UNDERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs. Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG\_INVALID\_FREQUENCY           The frequency is not supported in the active mode.
- DIAG\_INVALID\_DITHER              At least one dither parameter is invalid.

## SF\_HBridge

27333

**Function block type:** Safety function block (SF)

**Library:** ifmIOSafety.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))



This function block can continue to be used in existing applications for reasons of compatibility.

- Recommendation: In new applications, use the enhanced variant of this function block (SF\_[Type]\_Enh).

## Description

24136

The function block SF\_HBridge configures and controls a pair of output channels in the "HBridge" operating type to control a motor.

In addition to the function of the standard function block HBridge, the latter features safe deactivation of the output channel set at uiChannel1 by the S\_SafeControl input.



When the function block is called for the first time, both outputs (channel A and channel B) will be tested. The test can take several cycles until the H-bridge can be used (xPrepared = TRUE, if the block is ready).

## Input parameters

27247

Parameter	Data type	Description	Possible values	
S_SafeControl	SAFEBOOL	Safe deactivation of the output channel set at uiChannel.	FALSE	Initial value The output channel is in the safe state.
			TRUE	The output channel can be controlled via the uiValue.
xResetError	BOOL	Reset request for an occurring error	FALSE ⇒ TRUE	Reset request to the lower level system
			else	no action
uiChannel	UINT	1. Output channel (channel A) of the output channel pair	Group + channel → Data sheet → <b>Note on wiring</b> (→ p. <a href="#">39</a> )	
			Example:	
			106	Group 1 + channel 6 + 7
uiFrequency	UINT	PWM frequency of the output signal in [Hz]	→ Data sheet	
xDirection	BOOL	The direction in which the current flows via the bridge connections. Determines the direction of rotation of the connected motor.	FALSE	PWM Current Sourcing (CSO) is on channel A
			TRUE	PWM Current Sourcing (CSO) is on channel B
eBrakeMode	MODE_BRAKE	Brake mode that applies when the direction of rotation is changed or when stopping	→ <b>MODE_BRAKE (ENUM)</b> (→ p. <a href="#">307</a> )	
uiBrakeValue	UINT	Pulse/pause ratio of the PWM output signal at the corresponding current sinking output of the bridge in [%] The input is only relevant in the eBrakeModes that end with "_DYNAMIC" (= dynamic brake).	permissible = 0...1 000	
tBrakeTime	TIME	Indicates the braking time for the current sinking side of the bridge The input is only relevant in eBrakeModes ending with "_BTIME".	permissible = 0...1 h	
uiValue	UNIT	Pulse/pause ration of the PWM output signal in [%]	permissible = 0...1000	



## Output parameters

27087

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPrepared	BOOL	State of the FB outputs	FALSE	FB is still processed; FB outputs invalid
			TRUE	FB has been processed; FB outputs valid
uiOutCurrent	UINT	Measured current at the PWM output during normal operation in [mA] When braking, uiOutCurrent is = 0 because no regular current exists in the lowside path.	available = 0...final value of the measuring range	

Diagnostic codes (→ **Messages / diagnostic codes of the function blocks** (→ p. [463](#))):

- STAT\_PREPARING      State: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE      State: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_INACTIVE      State: FB/Function is inactive.
- ERR\_INTERNAL      Error: Internal system error
  - ▶ Contact the ifm Service Center!
- ERR\_UNDEFINED      Error: Unknown error
  - ▶ Contact the ifm Service Center!
- ERR\_INTERNAL\_COMMUNICATION      Error: Internal communication error
- ERR\_STUCK\_AT\_HIGH      Error: Signal frozen, signal state high.
- ERR\_OVERLOAD\_CURRENT      Error: Maximum current exceeded.
- ERR\_LOW\_CURRENT      Error: Minimum current not reached.
- DIAG\_INVALID\_VALUE      At least one input parameter is invalid or exceeds the permissible area.
- DIAG\_NO\_CALIB      The selected resource has no valid calibration.  
The displayed values are maybe faulty.
- DIAG\_ACCESS      FB/Function cannot access the required resource; The resource is not assigned to the accessing PLC in the IO mapping.
- DIAG\_STUCK\_AT\_HIGH      Error: Signal frozen, signal state high.
- DIAG\_OVERLOAD\_CURRENT      Maximum current exceeded.
- DIAG\_LOW\_CURRENT      Minimum current not reached.
- DIAG\_UNDERVOLTAGE\_VBBX
  - For inputs: Error: Reference voltage not reached.
  - For outputs: Error: The voltage of the corresponding output group supply or at VBB30 / VBB15 is not reached.
- DIAG\_INVALID\_FREQUENCY      The frequency is not supported in the active mode.
- DIAG\_INVALID\_DITHER      At least one dither parameter is invalid.

## 12.7.2 Library ifmPLCopenAddonSafe.library

### Content

SF_Equivalent_BOOL .....	375
SF_Antivalent_BOOL .....	377
SF_Equivalent_DINT .....	379
SF_Equivalent_UINT .....	382
SF_Equivalent_UDINT .....	385
SF_Equivalent_REAL .....	388
State chart SF_Equivalent_[Typ] .....	391

23971

The library `ifmPLCopenAddonSafe.library` provides validated and certified safety function blocks.

The included safety function blocks have the following properties:

- the FB input pairs expect standard data types
- the FB output signal is a safety data type

For this purpose, the inputs have 2 channels and are compared by means of the function block.

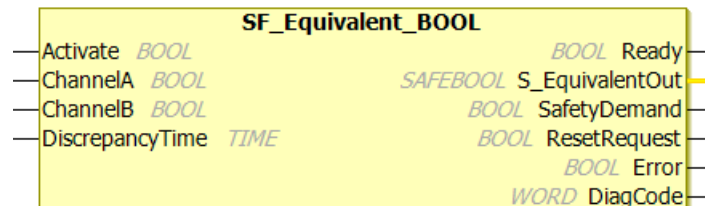
A safety output of the FB signals that the inputs have a valid status either within the monitoring time (SF\_Equivalent\_BOOL, SF\_Antivalent\_BOOL) or within a tolerance range (SF\_Equivalent\_UINT, SF\_Equivalent\_UDINT, SF\_Equivalent\_REAL).

More information: → User manual CODESYS Safety SIL2 V6.0 by CODESYS GmbH, §H2-6.2 (I/O devices without SAFE data types)

## SF\_Equivalent\_BOOL

27325

**Function block type:** Safety function block (SF)  
**Library:** ifmPLCopenAddonSafe.library  
**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG in the following chapter: → **Using the function blocks** (→ p. [207](#))

## Description

27138

The function block verifies the 2 input channels with standard data type BOOL (A and B) with regard to equivalence and provides the result at a SAFEBOOL output. For this, a discrepancy time that can be set is taken into consideration. The conditions at both input channels may not deviate for more than the set discrepancy time without an error being signalled.

## Input parameters

27236

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
ChannelA	BOOL	Input channel A	FALSE	Contact A open
			TRUE	Contact A closed
ChannelB	BOOL	Input channel B	FALSE	Contact B open
			TRUE	Contact B closed
DiscrepancyTime	TIME	Maximum monitoring time for the discrepancy check of both input channels.	permissible: T#0ms... T#49,71d	minimum time value maximum time value

## Output parameters

60615

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_EquivalentOut	SAFEBOOL	Safety-related output	FALSE	At least one input (A/B) is "inactive", or the last status change was outside of the discrepancy time.
			TRUE	Both inputs (A/B) are "active", and the last status change on both inputs was within the discrepancy time.
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

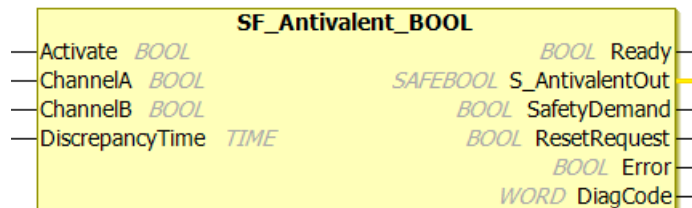
### Possible results for DIAGCODE:

Value [hex]	Description
0000	The function block is inactive. Start state.
8000	The function block is active and there is no error state. Safety output = TRUE
8001	The function block is active, safety output = FALSE. Initialisation.
8004	1st input = TRUE, waiting for 2nd input. Time monitoring has been started.
8014	1st input = TRUE, waiting for 2nd input. Time monitoring has been started.
8005	One channel = FALSE, waiting for the 2nd channel = FALSE. Time monitoring has been started.
C001	Error: Monitoring time has expired (FB waits for the 2nd channel)
C002	Error: Monitoring time has expired (FB waits for the 1st channel)
C003	Error: Monitoring time has expired (FB waits for both channels = inactive)

## SF\_Antivalent\_BOOL

27317

**Function block type:** Safety function block (SF)  
**Library:** ifmPLCopenAddonSafe.library  
**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### Description

27139

The function block verifies the 2 input channels with standard data type BOOL (A and B) with regard to antivalence and provides the result at a SAFEBOOL output. For this, a discrepancy time that can be set is taken into consideration.

The conditions at both input channels may not be identical for more than the set discrepancy time without an error being signalled.

The input channel A monitors the NC contact (NC = normally closed).

The input channel B monitors the NO contact (NO = normally open).

### Input parameters

27237

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
ChannelA	BOOL	Input channel A	FALSE	Contact A open
			TRUE	Contact A closed
ChannelB	BOOL	Input channel B	FALSE	Contact B open
			TRUE	Contact B closed
DiscrepancyTime	TIME	Maximum monitoring time for the discrepancy check of both input channels.	permissible: T#0ms... T#49,71d	minimum time value maximum time value

## Output parameters

27110

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_AntivalentOut	SAFEBOOL	Safety-related output	FALSE	At least one input (A/B) is "inactive", or the last status change was outside of the discrepancy time.
			TRUE	Both inputs (A/B) are "active", and the last status change on both inputs was within the discrepancy time.
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

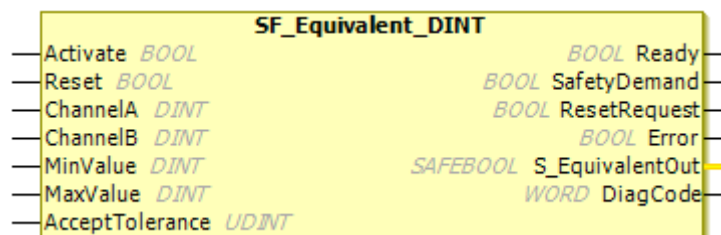
### Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8001	The function block is active, safety output = FALSE. Initialisation.
8000	The function block is active and there is no error state. Safety output = TRUE
8004	1st input = TRUE, waiting for 2nd input. Time monitoring has been started.
8014	1st input = TRUE, waiting for 2nd input. Time monitoring has been started.
8005	One channel = inactive, waiting for the other channel = inactive. Time monitoring has been started.
C001	Error: Monitoring time has expired (FB waits for the 2nd channel)
C002	Error: Monitoring time has expired (FB waits for the 1st channel)
C003	Error: Monitoring time has expired (FB waits for both channels = inactive)

## SF\_Equivalent\_DINT

60618

**Function block type:** Safety function block (SF)  
**Library:** ifmPLCopenAddonSafe.library  
**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

60619

This function block evaluates 2 DINT data type input channels and compares whether the read values are within the defined tolerance and within the defined value range.

60619

The function block's behaviour is based on the PLCopen safe function blocks according to the PLCopen specification TC5 Safety part 1. The function block features the extended outputs according to the PLCopen specification TC5 Safety part 3, chapter 2.4.

## Input parameters

60620

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
ChannelA	DINT	Input channel A	0	Initial value
ChannelB	DINT	Input channel B	0	Initial value
MinValue	DINT	Permissible minimum value on the inputs ChannelA and ChannelB.	-2147483648...2147483647	Initial value Value range DINT
MaxValue	DINT	Permissible maximum value on the inputs ChannelA and ChannelB.	-2147483648...2147483647	Initial value Value range DINT
AcceptTolerance	DINT	Permissible deviation between the values of the inputs ChannelA and ChannelB	-2147483648...2147483647	Permissible values
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

24095

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_EquivalentOut	SAFEBOOL	Safety-related output	FALSE	The absolute difference between the values read at the inputs A/B is outside of the set tolerance value or outside of the set value range.
			TRUE	The absolute difference between the values read at the inputs A/B is within the set tolerance value and within the set value range
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested



## Error codes

60627

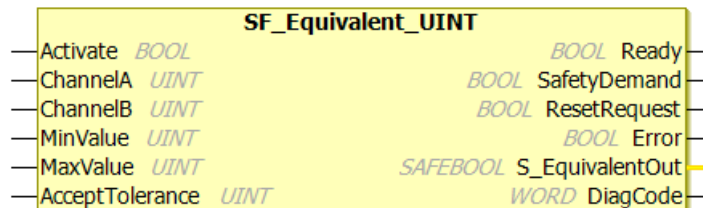
Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8001	The function block is active, safety output = FALSE. Initialisation.
8000	The function block is active and there is no error state. Safety output = TRUE
C001	Error: <ul style="list-style-type: none"> <li>▪ UINT: Parameter AcceptTolerance &gt; 65534 (UINT upper limit value range - 1);</li> <li>▪ UDINT: Parameter AcceptTolerance &gt; 4294967294 (UDINT upper limit value range - 1);</li> <li>▪ DINT: Parameter AcceptTolerance &gt; 2147483646 (DINT upper limit value range - 1);</li> <li>▪ REAL: Parameter AcceptTolerance &gt;= 3.402823e+32 (1/1000000 of the maximum REAL value that can be visualised);</li> </ul> Safety output = FALSE
C002	Error: MinValue > MaxValue; Safety output = FALSE
C003	Error: ChannelA > MaxValue; Safety output = FALSE
C004	Error: ChannelA < MinValue; Safety output = FALSE
C005	Error: ChannelB > MaxValue; Safety output = FALSE
C006	Error: ChannelB < MinValue; Safety output = FALSE
C007	Error: ChannelB > ChannelA + tolerance; Safety output = FALSE
C008	Error: ChannelB < ChannelA - tolerance; Safety output = FALSE

## SF\_Equivalent\_UINT

27328

**Function block type:** Safety function block (SF)  
**Library:** ifmPLCopenAddonSafe.library  
**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27140

This function block evaluates 2 UINT data type input channels and compares whether the read values are within the defined tolerance and within the defined value range.

27140

The function block's behaviour is based on the PLCopen safe function blocks according to the PLCopen specification TC5 Safety part 1. The function block features the extended outputs according to the PLCopen specification TC5 Safety part 3, chapter 2.4.

## Input parameters

27238

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
ChannelA	UINT	Input channel A	0	Initial value
ChannelB	UINT	Input channel B	0	Initial value
MinValue	UINT	Permissible minimum value on the inputs ChannelA and ChannelB.	0 0...65 535	Initial value permissible values
MaxValue	UINT	Permissible maximum value on the inputs ChannelA and ChannelB.	0 0...65 535	Initial value permissible values
AcceptTolerance	UINT	Permissible deviations between the values of the inputs ChannelA and ChannelB	0...65 534	permissible values
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27111

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_EquivalentOut	SAFEBOOL	Safety-related output	FALSE	The absolute difference between the values read at the inputs A/B is outside of the set tolerance value or outside of the set value range.
			TRUE	The absolute difference between the values read at the inputs A/B is within the set tolerance value and within the set value range
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested

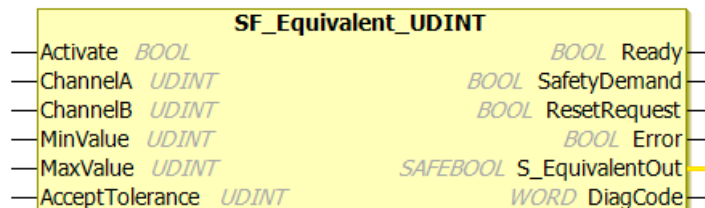
## Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8001	The function block is active, safety output = FALSE. Initialisation.
8000	The function block is active and there is no error state. Safety output = TRUE
C001	Error: <ul style="list-style-type: none"> <li>▪ UINT: Parameter AcceptTolerance &gt; 65534 (UINT upper limit value range - 1);</li> <li>▪ UDINT: Parameter AcceptTolerance &gt; 4294967294 (UDINT upper limit value range - 1);</li> <li>▪ DINT: Parameter AcceptTolerance &gt; 2147483646 (DINT upper limit value range - 1);</li> <li>▪ REAL: Parameter AcceptTolerance &gt;= 3.402823e+32 (1/1000000 of the maximum REAL value that can be visualised);</li> </ul> Safety output = FALSE
C002	Error: MinValue > MaxValue; Safety output = FALSE
C003	Error: ChannelA > MaxValue; Safety output = FALSE
C004	Error: ChannelA < MinValue; Safety output = FALSE
C005	Error: ChannelB > MaxValue; Safety output = FALSE
C006	Error: ChannelB < MinValue; Safety output = FALSE
C007	Error: ChannelB > ChannelA + tolerance; Safety output = FALSE
C008	Error: ChannelB < ChannelA - tolerance; Safety output = FALSE

## SF\_Equivalent\_UDINT

27327

**Function block type:** Safety function block (SF)  
**Library:** ifmPLCopenAddonSafe.library  
**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### Description

27141

This function block evaluates 2 UDINT data type input channels and compares whether the read values are within the defined tolerance and within the defined value range.

27141

The function block's behaviour is based on the PLCopen safe function blocks according to the PLCopen specification TC5 Safety part 1. The function block features the extended outputs according to the PLCopen specification TC5 Safety part 3, chapter 2.4.

### Input parameters

27239

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
ChannelA	UDINT	Input channel A	0	Initial value
ChannelB	UDINT	Input channel B	0	Initial value
MinValue	UDINT	Permissible minimum value on the inputs ChannelA and ChannelB.	0...4294967295	Initial value Value range UDINT
MaxValue	UDINT	Permissible maximum value on the inputs ChannelA and ChannelB.	0...4294967295	Initial value Value range UDINT
AcceptTolerance	UDINT	Permissible deviation between the values of the inputs ChannelA and ChannelB	0...4294967294	permissible values
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27111

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_EquivalentOut	SAFEBOOL	Safety-related output	FALSE	The absolute difference between the values read at the inputs A/B is outside of the set tolerance value or outside of the set value range.
			TRUE	The absolute difference between the values read at the inputs A/B is within the set tolerance value and within the set value range
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested

## Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8001	The function block is active, safety output = FALSE. Initialisation.
8000	The function block is active and there is no error state. Safety output = TRUE
C001	Error: <ul style="list-style-type: none"> <li>▪ UINT: Parameter AcceptTolerance &gt; 65534 (UINT upper limit value range - 1);</li> <li>▪ UDINT: Parameter AcceptTolerance &gt; 4294967294 (UDINT upper limit value range - 1);</li> <li>▪ DINT: Parameter AcceptTolerance &gt; 2147483646 (DINT upper limit value range - 1);</li> <li>▪ REAL: Parameter AcceptTolerance &gt;= 3.402823e+32 (1/1000000 of the maximum REAL value that can be visualised);</li> </ul> Safety output = FALSE
C002	Error: MinValue > MaxValue; Safety output = FALSE
C003	Error: ChannelA > MaxValue; Safety output = FALSE
C004	Error: ChannelA < MinValue; Safety output = FALSE
C005	Error: ChannelB > MaxValue; Safety output = FALSE
C006	Error: ChannelB < MinValue; Safety output = FALSE
C007	Error: ChannelB > ChannelA + tolerance; Safety output = FALSE
C008	Error: ChannelB < ChannelA - tolerance; Safety output = FALSE

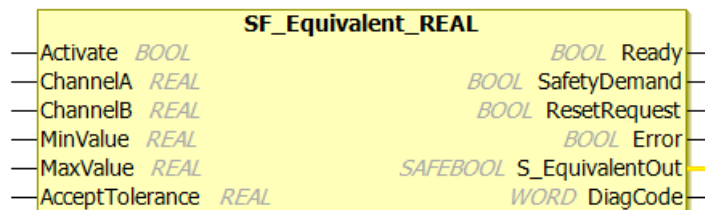
## SF\_Equivalent\_REAL

27326

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenAddonSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### Description

27142

This function block evaluates 2 REAL data type input channels and compares whether the measured values are within the defined tolerance and within the defined value range.

27142

The function block's behaviour is based on the PLCopen safe function blocks according to the PLCopen specification TC5 Safety part 1. The function block features the extended outputs according to the PLCopen specification TC5 Safety part 3, chapter 2.4.

### Input parameters

27240

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
ChannelA	REAL	Input channel A	0.0	Initial value
ChannelB	REAL	Input channel B	0.0	Initial value
MinValue	REAL	Permissible minimum value on the inputs ChannelA and ChannelB.	0.0 0.0 ... 3.402823E+38	Initial value permissible values
MaxValue	REAL	Permissible maximum value on the inputs ChannelA and ChannelB.	0.0 0.0 ... 3.402823E+38	Initial value permissible values
AcceptTolerance	REAL	Permissible deviation between the values of the inputs ChannelA and ChannelB	0.0 ... 3.402823e+32	permissible values Upper limit: 1/1000000 of the maximum REAL value that can be visualised.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error



## Output parameters

27111

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_EquivalentOut	SAFEBOOL	Safety-related output	FALSE	The absolute difference between the values read at the inputs A/B is outside of the set tolerance value or outside of the set value range.
			TRUE	The absolute difference between the values read at the inputs A/B is within the set tolerance value and within the set value range
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested

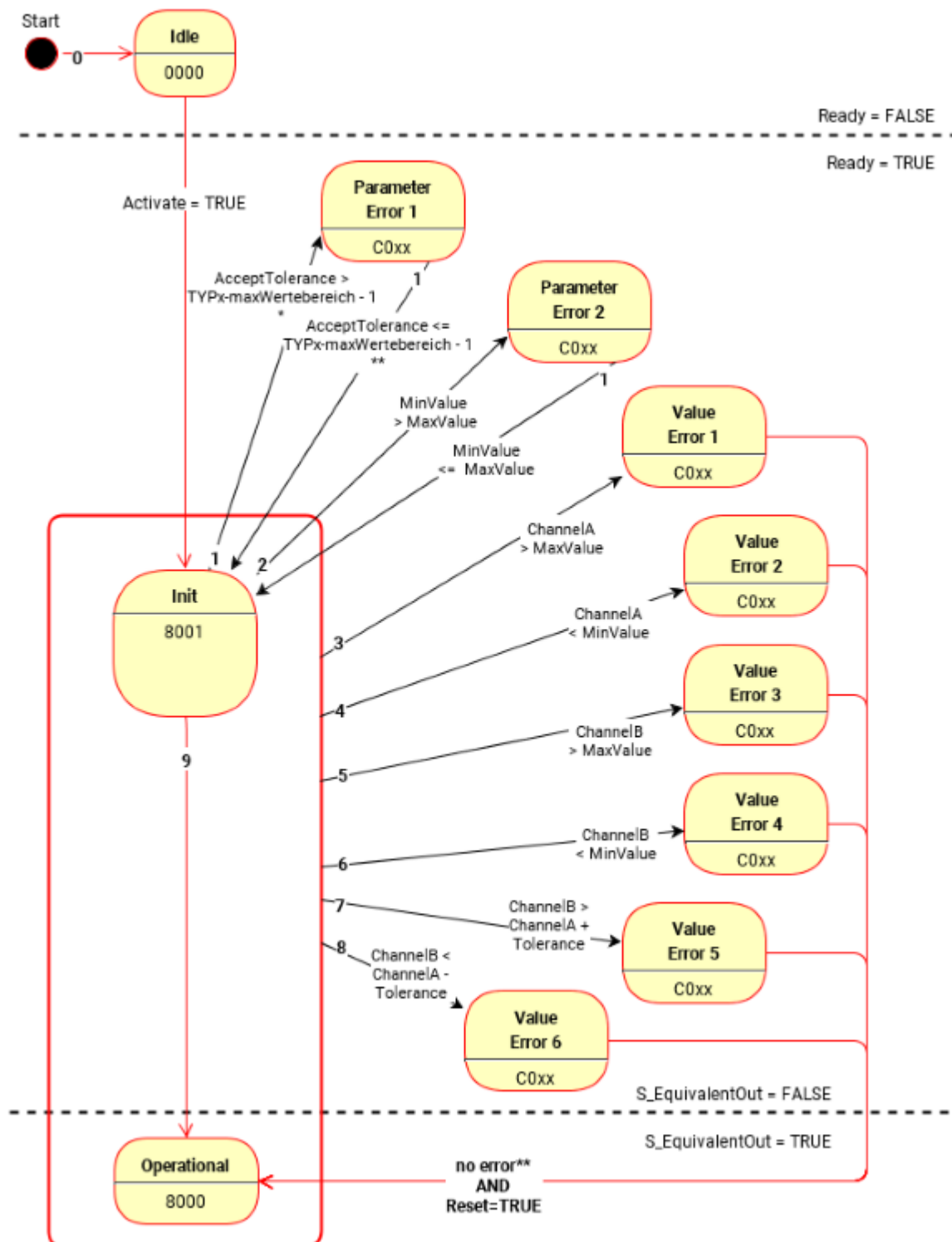
## Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8001	The function block is active, safety output = FALSE. Initialisation.
8000	The function block is active and there is no error state. Safety output = TRUE
C001	Error: <ul style="list-style-type: none"> <li>▪ UINT: Parameter AcceptTolerance &gt; 65534 (UINT upper limit value range - 1);</li> <li>▪ UDINT: Parameter AcceptTolerance &gt; 4294967294 (UDINT upper limit value range - 1);</li> <li>▪ DINT: Parameter AcceptTolerance &gt; 2147483646 (DINT upper limit value range - 1);</li> <li>▪ REAL: Parameter AcceptTolerance &gt;= 3.402823e+32 (1/1000000 of the maximum REAL value that can be visualised);</li> </ul> Safety output = FALSE
C002	Error: MinValue > MaxValue; Safety output = FALSE
C003	Error: ChannelA > MaxValue; Safety output = FALSE
C004	Error: ChannelA < MinValue; Safety output = FALSE
C005	Error: ChannelB > MaxValue; Safety output = FALSE
C006	Error: ChannelB < MinValue; Safety output = FALSE
C007	Error: ChannelB > ChannelA + tolerance; Safety output = FALSE
C008	Error: ChannelB < ChannelA - tolerance; Safety output = FALSE

## State chart SF\_Equivalent\_[Typ]

27358

State chart for the safety function blocks SF\_Equivalent\_DINT, SF\_Equivalent\_UINT, SF\_Equivalent\_UDINT, SF\_Equivalent\_REAL:





For the FB\_Equivalent\_REAL, the following conditions apply due to the REAL values for the transitions to / from the Parameter Error 1 status:

\*  $\text{AcceptTolerance} > \text{maximum value REAL} / 1000000 = 3.402823\text{e}+32$

\*\*  $\text{AcceptTolerance} \leq \text{maximum value REAL} / 1000000 = 3.402823\text{e}+32$

In addition to the diagram, the following conditions apply:

- The transition from any state into the Idle state through Activate = FALSE is not shown here for the sake of clarity. This transition always has priority 0 = highest priority.
- If the input Reset=TRUE and there is no other error, the function block changes from the Value Error x state to the Operational state.
- If there is a Value Error x, an error of higher priority occurs, the function block changes to the higher error state Value Error x.
- If the input Reset=TRUE and one or several other errors are active, the function block changes to this Value Error x according to the priority.
- Lower error priority = higher priority value, high error priority= low priority value

## 12.7.3 Library ifmPLCopenSafe.library

### Content

SF_Antivalent .....	394
SF_CamshaftMonitor .....	396
SF_DoubleValveMonitoring .....	400
SF_EDM .....	404
SF_EmergencyStop .....	408
SF_EnableSwitch .....	412
SF_Equivalent .....	415
SF_ESPE .....	417
SF_FootSwitch .....	421
SF_GuardLocking .....	424
SF_GuardMonitoring .....	428
SF_ModeSelector .....	432
SF_OutControl .....	435
SF_SafetyRequest .....	439
SF_SingleValveCycleMonitoring .....	442
SF_SingleValveMonitoring .....	446
SF_TwoHandControlTypeII .....	450
SF_TwoHandControlTypeIII .....	453
SF_ValveGroupControl .....	456

27156

This library provides validated and certified safety function blocks according to PLCopen specifications.

The function blocks are extended by the outputs SafetyDemand and ResetRequest according to PLCopen specification TC5 Safety Part 3, Chapter 2.4 .

Details and possible deviations are described in the following chapters.

27156



The project engineer is responsible for the use of the safety PLCopen function blocks provided in CODESYS.

Detailed information about the specifications: [www.plcopen.org](http://www.plcopen.org) > PLCopen Safety

## SF\_Antivalent

27316

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27120

The function block checks 2 SAFEBOOL input channels (S\_ChannelNC and S\_ChannelNO) with regard to antivalence and provides the result at the SAFEBOOL output S\_AntivalentOut.

For this, a discrepancy time that can be set is taken into consideration.

The conditions at both input channels may not be identical for more than the set discrepancy time without an error being signalled.

## Input parameters

27218

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_ChannelNC	SAFEBOOL	Input channel NC contact (Normally Closed)	FALSE	NC contact open. Channel is inactive. (0 V)
			TRUE	NC contact closed. Channel is active. (24 V)
S_ChannelNO	SAFEBOOL	Input channel NO contact (Normally Open)	FALSE	NO contact open. Channel is active. (0 V)
			TRUE	NO contact closed. Channel is inactive. (24 V)
DiscrepancyTime	TIME	Maximum monitoring time for the discrepancy check of both input channels.	permissible: T#0ms... T#49,71d	minimum time value maximum time value

## Output parameters

27091

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_AntivalentOut	SAFEBOOL	Safety-related output	FALSE	At least one input (A/B) is "inactive", or the last status change was outside of the discrepancy time.
			TRUE	Both inputs (A/B) are "active", and the last status change on both inputs was within the discrepancy time.
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

### Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8001	The function block is active, safety output = FALSE. Initialisation.
8000	The function block is active and there is no error state. Safety output = TRUE
8004	S_ChannelNC = TRUE, waiting for S_ChannelNO = FALSE. Time monitoring has been started.
8014	S_ChannelNO = FALSE, waiting for S_ChannelNC = TRUE. Time monitoring has been started.
8005	One channel = inactive, waiting for the other channel = inactive. Time monitoring has been started.
C001	Error: Monitoring time expired (FB waits for S_ChannelNO = FALSE)
C002	Error: Monitoring time expired (FB waits for S_ChannelNC = TRUE)
C003	Error: Monitoring time expired since one channel = inactive (FB waits for the other channel = inactive)

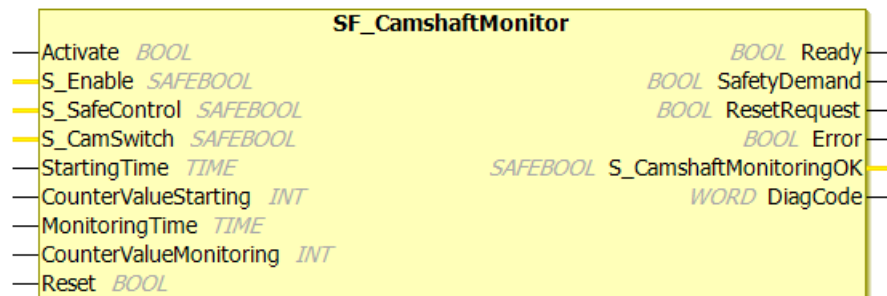
## SF\_CamshaftMonitor

27318

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27137

The function block monitors the camshaft. It monitors a defined number of signal changes for a set duration.



## Input parameters

27235

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_Enable	SAFEBOOL	Control signal to deactivate the monitoring function.	FALSE	Initial value Camshaft monitoring not active
			TRUE	Camshaft monitoring active
S_SafeControl	SAFEBOOL	Input signal Indicates that the press is moving and that monitoring must be activated.	FALSE	Initial value Press stands still.
			TRUE	Press is moving.
S_CamSwitch	SAFEBOOL	Input signal Signal of the cam switch.	FALSE	Initial value Cam switch off
			TRUE	Cam switch on
StartingTime	TIME	Starting time of the machine in [ms]. Duration in which the value of CounterValueStarting must be reached.	T#0ms	Initial value
			0... 4294967295 [ms]	Value range: 4294967295 ms = T#49.71d
CounterValueStarting	INT	Number of signal changes that must be reached during machine start-up.	0	Initial value
			0...40	Value range
MonitoringTime	TIME	Monitoring time during cycle operation in [ms] Within this time, the signal change value set at CounterValueMonitoring must be reached. The MonitoringTime must be shorter than the StartingTime.	T#0ms	Initial value
			0... 4294967295 [ms]	Value range: 4294967295 ms = T#49.71d
CounterValueMonitoring	INT	Number of signal changes that must be reached within the MonitoringTime during cyclic machine operation.	0	Initial value
			0...40	Value range
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27108

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_CamshaftMonitoringOK	SAFEBOOL	Output signal Indicates camshaft monitoring status.	FALSE	Initial value Camshaft monitoring not ok or signal error!
			TRUE	Camshaft monitoring sequence ok or not activated.
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

## Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	The function block is active and there is no error state. Safety output = TRUE
8001	The function block is active, safety output = FALSE. Initialisation.
8100	Machine is in the start stage.
8200	Machine is in cyclic operation.
8300	Camshaft monitoring deactivated.
C000	Error: Monitoring time longer than start time.
C001	Error: Reset permanently TRUE in the Init (8001) state. (Only in case of manual reset.)
C010	Error: Counter values invalid.
C011	Error: Reset permanently TRUE in the Counter Error Starting (C400) state.
C021	Error: Reset permanently TRUE in the Counter Error Cyclic (C410) state.
C400	Error: Number of signal changes during the start time too low or number of signal changes greater than 50. Reset required.
C410	Error: Number of signal changes during the monitoring time too low or number of signal changes greater than 50. Reset required.

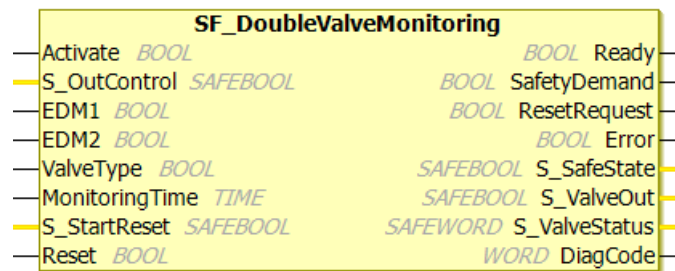
## SF\_DoubleValveMonitoring

27320

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### description

27135

The function block monitors the switching behaviour of 2 fluidic safety valves / double valves (safety valves for presses).

The safe state is monitored by static feedback signals of the valves (by monitoring the coil position). The value in MonitoringTime is used to monitor the feedback signal.

24885

The function block's reset behaviour can be set via the input S\_StartReset.



### WARNING

For **PowerOn** and **S\_StartReset = TRUE**, the function block starts operating without acknowledging the start.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.

## Input parameters

27233

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_OutControl	SAFEBOOL	safe state	FALSE	Initial value The valves must be in the safe state.
			TRUE	The valves must be in the non safe state.
EDM1	BOOL	Valve 1 feedback	FALSE	Initial value Significance depends on the ValveType input.
			TRUE	Significance depends on the ValveType input.
EDM2	BOOL	Valve 2 feedback	FALSE	Initial value Significance depends on the ValveType input.
			TRUE	Significance depends on the ValveType input.
ValveType	BOOL	Parameter to set the polarity of the valve feedback signal for the safe state.	FALSE	Initial value The safe state of the valve must be reported back with a LOW signal to EDM. In the non safe state, there must be a HIGH signal at EDM.
			TRUE	The safe state of the valve must be reported back with a HIGH signal to EDM. In the non safe state, there must be a LOW signal at EDM.
MonitoringTime	TIME	Monitoring time in [ms] Within this time, the feedback signals must have reported back the switching of the valve.	T#0ms	Initial value
S_StartReset	SAFEBOOL	Selection manual / automatic reset when the function block is activated.  Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	FALSE	(initial value) When the function block has been activated, manual reset is required.
			TRUE	When the function block has been activated, an automatic reset will take place.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27106

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_SafeState	SAFEBOOL	Safe state of the valve / valves	FALSE	Initial value The valve is not in the safe state or an error has occurred.
			TRUE	The valve is in the safe state and no error has occurred.
S_ValveOut	SAFEBOOL	Safety-related output to control the connected safety valves.	FALSE	Initial value Deactivate connected valves
			TRUE	Activate connected valves
S_ValveStatus	SAFEWORD	Status signal of the safety valve This output can be used for connection with the function block SF_ValveGroupControl.	16#0000	Initial value
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

### Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
83FF	Log on to valve group.
8401	Function block activation. Reset required.
8802	Waiting for feedback signals (EDM) to confirm the safe coil position of the valves. The monitoring time starts when entering the state.
8804	The feedback signals (EDM) report back that the valves are in the safe state.
8010	The feedback signals (EDM) report back that the valves are in the switched state.
8020	Waiting for the feedback signals (EDM) to confirm the switched coil position of the valves. The timer starts when entering the state.

Value [hex]	Description
C100	Error: Identical signal curve detected at S_OutControl and Reset (rising edge FALSE => TRUE in the same cycle) during manual reset. Could be a programming error.
C001	Error: Reset permanently TRUE in the Init (8401) state during manual reset.
C0n1	Error: Reset permanently TRUE in the EDM Error Xn0 condition (CXn0). With n=1..9
Cx10	Feedback error; the EDM1 signal is invalid in the 8802 state. EDM1 = FALSE, when activating the safety function.
Cx20	Feedback error; the signal EDM2 is invalid in the 8802 state. EDM2 = FALSE, when activating the safety function.
Cx30	Feedback error; the signals 1 EDM1 and EDM2 are invalid in the 8802 state. EDM1 = FALSE and EDM2 = FALSE, when activating the safety function.
Cx40	Feedback error; the signal EDM1 is invalid in the states 8802 and 8804: In 8802, EDM1 is = FALSE and the monitoring time has expired. In 8804, EDM1 becomes = FALSE without the valve having received a switch command.
Cx50	Feedback error; the signal EDM2 is invalid in the states 8802 and 8804: In 8802, EDM2 is = FALSE and the monitoring time has expired. In 8804, EDM2 becomes = FALSE without the valve having received a switch command.
Cx60	Feedback error; the signals EDM1 and EDM2 are invalid in the states 8802 and 8804: In 8802, EDM1 is = FALSE and EDM2 is = FALSE and the monitoring time has expired. In 8804, EDM1 is = FALSE and EDM2 is = FALSE, without the valves having received any switch command.
Cx70	Feedback error; the signal EDM1 is invalid in state 8010 or 8020: In 8020, EDM1 is = TRUE and the monitoring time has expired or S_OutControl has been set to FALSE before EDM1 had confirmed the switching. In 8010, EDM1 is = TRUE, without the valve having received a switching command.
Cx80	Feedback error; the signal EDM2 is invalid in the state 8010 or 8020: In 8020, EDM2 is = TRUE and the monitoring time has expired or S_OutControl has been set to FALSE before EDM2 had confirmed the switching. In 8010, EDM2 is = TRUE without the valve having received any switch command.
Cx90	Feedback error; the signals EDM1 and EDM2 are invalid in the state 8010 or 8020: In 8020, EDM1 is = TRUE and EDM2 is = TRUE and the monitoring time has expired or S_OutControl has been set to FALSE before EDM1 and EDM2 have confirmed the switching. In 8010, EDM1 is = TRUE and EDM2 is = TRUE without the valves having received any switch command.

## SF\_EDM

27321

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27131

The function block SF\_EDM (EDM = **E**xternal **D**evice **M**onitoring = monitoring of external devices) controls a safety output and monitors the controlled actuators. For this, a monitoring time can be set.

24885

The function block's reset behaviour can be set via the input S\_StartReset.



### WARNING

For **PowerOn** and **S\_StartReset = TRUE**, the function block starts operating without acknowledging the start.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.



## Input parameters

27229

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_OutControl	SAFEBOOL	Control signal Can be connected with other function blocks, e.g. SF_OutControl, SF_TwoHandControlTypell, etc.	FALSE	Initial value Safety output deactivated
			TRUE	Safety output active
S_EDM1	SAFEBOOL	Feedback signal of the first connected actuator	FALSE	Initial value Switching state of the first connected actuator
			TRUE	Initial state of the first connected actuator
S_EDM2	SAFEBOOL	Feedback signal of the second connected actuator	FALSE	Initial value Switching state of the second connected actuator
			TRUE	Initial state of the second connected actuator
MonitoringTime	TIME	Monitoring time in [ms] Within this time, the feedback signals must have reported back the switching of the connected actuators.	T#0ms	Initial value
S_StartReset	SAFEBOOL	Selection manual / automatic reset when the function block is activated.  Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	FALSE	(initial value) When the function block has been activated, manual reset is required.
			TRUE	When the function block has been activated, an automatic reset will take place.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27102

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_EDM_Out	SAFEBOOL	Safety-related output to control the connected actuator	FALSE	Initial value Switch off connected actuator
			TRUE	Switch on connected actuator
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

### Possible results for DIAGCODE:

Value [hex]	Description
8001	Start disable for function block activation is active. Reset required.
8010	EDM control is not active (S_EDM_Out = FALSE). Monitoring active.
8000	EDM control is active (S_EDM_Out = TRUE). Monitoring active.
C001	Error: Reset is permanently TRUE in the Init (8001) state. (Only in case of S_StartReset = FALSE, manual reset.)
C011	Error: Reset permanently TRUE or identical signal curve at EDM1 and Reset (rising edge simultaneous at EDM1 and Reset) in the EDM Error 11 (C010) state.
C021	Error: Reset permanently TRUE or identical signal curve at EDM2 and Reset (rising edge simultaneous at EDM2 and Reset) in the EDM Error 12 (C020) state.
C031	Error: Reset permanently TRUE or identical signal curve at EDM1, EDM2 and Reset (rising edge simultaneous at EDM1, EDM2 and Reset) in the EDM Error 13 (C030) state.
C041	Error: Reset permanently TRUE or identical signal curve at EDM1 and Reset (rising edge simultaneous at EDM1 and Reset) in the EDM Error 21 (C040) state.
C051	Error: Reset permanently TRUE or identical signal curve at EDM2 and Reset (rising edge simultaneous at EDM2 and Reset) in the EDM Error 22 (C050) state.
C061	Error: Reset permanently TRUE or identical signal curve at EDM1, EDM2 and Reset (rising edge simultaneous at EDM1, EDM2 and Reset) in the EDM Error 23 (C060) state.
C071	Error: Reset permanently TRUE in the EDM Error 31 (C070) state.

Value [hex]	Description
C081	Error: Reset permanently TRUE in the EDM Error 32 (C080) state.
C091	Error: Reset permanently TRUE in the EDM Error 33 (C090) state.
C010	Error: The signal at EDM1 is invalid in the initial actuator state. In state 8010, the EDM1 signal is FALSE if S_OutControl is being activated.
C020	Error: The signal at EDM2 is invalid in the initial actuator state. In state 8010, the EDM2 signal is FALSE if S_OutControl is being activated.
C030	Error: The signals at EDM1 and EDM2 are invalid in the initial actuator state. In state 8010, the signals EDM1 and EDM2 are FALSE if S_OutControl is being activated.
C040	Error: The signal at EDM1 is invalid in the initial actuator state. In state 8010, the EDM1 signal is FALSE and the monitoring time has elapsed.
C050	Error: The signal at EDM2 is invalid in the initial actuator state. In state 8010, the EDM2 signal is FALSE and the monitoring time has elapsed.
C060	Error: The signals at EDM1 and EDM2 are invalid in the initial actuator state. In the 8010 state, the signals EDM1 and EDM2 are FALSE and the monitoring time has elapsed.
C070	Error: The signal at EDM1 is invalid in the initial actuator state. In state 8000, the EDM1 signal is TRUE and the monitoring time has elapsed.
C080	Error: The signal at EDM2 is invalid in the initial actuator state. In state 8000, the EDM2 signal is TRUE and the monitoring time has elapsed.
C090	Error: The signals at EDM1 and EDM2 are invalid in the initial actuator state. In state 8000, the signals EDM1 and EDM2 are TRUE and the monitoring time has elapsed.
C111	Error: Identical signal curve detected at S_OutControl and Reset (positive edge in the same cycle). Possibly programming error. (Only in case of S_StartReset = FALSE, manual reset.)

## SF\_EmergencyStop

27322

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### Description

27122

The function block SF\_EmergencyStopP monitors an E-stop.

24785

The reset behaviour of the function block can be set via the inputs S\_StartReset and S\_AutoReset.



### WARNING

For **PowerOn** and **S\_StartReset = TRUE**, the function block starts operating without acknowledging the start.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.



### WARNING

When the **protective equipment is reset** and **S\_AutoReset = TRUE**, the function block starts operating without manual acknowledgement.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.

## Input parameters

27220

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_EStopIn	SAFEBOOL	Input signal of the safety switch (e.g. E-stop)	FALSE	(initial value) Safety switch (e.g. E-stop) has been engaged.
			TRUE	Safety switch (e.g. E-stop) has not been engaged.
S_StartReset	SAFEBOOL	Selection manual / automatic reset when the function block is activated.	FALSE	(initial value) When the function block has been activated, manual reset is required.
		Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	TRUE	When the function block has been activated, an automatic reset will take place.
S_AutoReset	SAFEBOOL	Selection of manual / automatic reset after the protective equipment has been reset (e.g. E-stop, ESPE, etc.).	FALSE	(initial value) After resetting the protective equipment, manual reset is required.
		Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	TRUE	After resetting the protective equipment, an automatic reset takes place.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27093

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_EStopOut	SAFEBOOL	Output for safety-related feedback.	FALSE	(initial value): Safety-related output is deactivated: • E-stop is engaged • a reset is required • or there is an internal error
			TRUE	Safety-related output is active
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

### Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8001	Activation = TRUE, the FB is active. Verification whether a manual reset is required.
8002	Function block is ready: Activation = TRUE Function block waits for safety-related input=TRUE
8003	Function block is active: Activate = TRUE, safety input = TRUE FB waits for edge to reset = FALSE => TRUE
8004	Activate = TRUE Safety request detected Verify if S_AutoReset = TRUE or FALSE Function block waits for safety-related input=TRUE
8005	Activation = TRUE Safety-related input=TRUE FB waits for the rising edge to reset (FALSE => TRUE) or for S_AUTORESET=TRUE

Value [hex]	Description
8000	Activation = TRUE Safety-related input=TRUE Safety output = TRUE
C001	Error because permanently reset = TRUE in 8003, waiting for rest = FALSE
C002	Error because permanently reset = TRUE in 8005, waiting for rest = FALSE

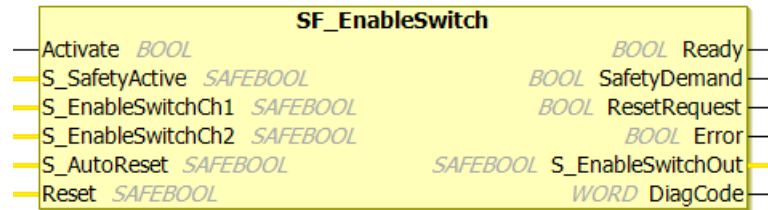
## SF\_EnableSwitch

27323

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27128

The function block SF\_EnableSwitch is used to evaluate the safe signals of a release switch with three switching stages.

25160

The function block's reset behaviour can be set via the input S\_AutoReset.



### WARNING

When the **protective equipment is reset** and **S\_AutoReset = TRUE**, the function block starts operating without manual acknowledgement.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.



## Input parameters

27226

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_SafetyActive	SAFEBOOL	Indicates if the safe mode is active or not.	FALSE	Initial value Safe mode is inactive.
			TRUE	Safe mode is active.
S_EnableSwitchCh1	SAFEBOOL	Signal of the contacts E1 and E2 of the connected release switch.	FALSE	Initial value Connected switching contacts are open
			TRUE	Connected switching contacts are closed
S_EnableSwitchCh2	SAFEBOOL	Signal of the contacts E4 and E4 of the connected release switch.	FALSE	Initial value Connected switching contacts are open
			TRUE	Connected switching contacts are closed
S_AutoReset	SAFEBOOL	Selection of manual / automatic reset after the protective equipment has been reset (e.g. E-stop, ESPE, etc.).  Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	FALSE	(initial value) After resetting the protective equipment, manual reset is required.
			TRUE	After resetting the protective equipment, an automatic reset takes place.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27099

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_EnableSwitchOut	SAFEBOOL	Safety-related output	FALSE	Initial value Safety-related release output is inactive
			TRUE	Safety-related release output is active
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

### Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8004	The safety-related release output is inactive (FALSE).
8005	The safety-related release output is active (TRUE).
8006	The safety-related release output is active (TRUE) and the switch is in position 1.
8007	The safety-related release output is active (TRUE) and the switch is in position 3.
8000	The safety-related release output is active (TRUE) and the switch is in position 2.
C001	Error: Reset permanently TRUE in the Operation Error 2 (C020) state
C002	Error: Reset permanently TRUE in the Operation Error 4 (C040) state
C010	Error: Release switch in position 1 while S_SafetyActive is being activated.
C020	Error: Release switch in position 1 after Operation Error 1 (C010).
C030	Error: Release switch in position 2 after position 3.
C040	Error: release switch NOT in position 3 after Operation Error 3 (C030).

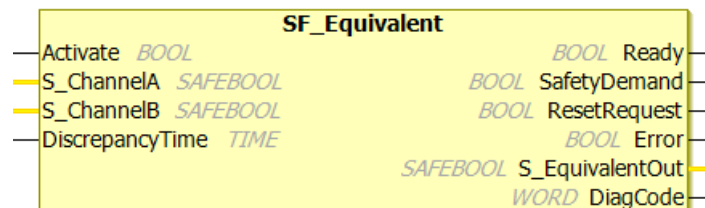
## SF\_Equivalent

27324

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27119

The function block checks 2 SAFEBOOL input channels (S\_ChannelA and S\_ChannelB) with regard to equivalence and provides the result at the SAFEBOOL output S\_EquivalentOut.

For this, a discrepancy time that can be set is taken into consideration.

The conditions at both input channels may not deviate for more than the set discrepancy time without an error being signalled.

## Input parameters

27217

Parameters	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_ChannelA	SAFEBOOL	Input channel A	FALSE	Contact A open
			TRUE	Contact A closed
S_ChannelB	SAFEBOOL	Input channel B	FALSE	Contact B open
			TRUE	Contact B closed
DiscrepancyTime	TIME	Maximum monitoring time for the discrepancy check of both input channels.	permissible: T#0ms... T#49,71d	minimum time value maximum time value

## Output parameters

27090

Parameters	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_EquivalentOut	SAFEBOOL	Safety-related output	FALSE	One input (A/B) = FALSE or both inputs = TRUE, but the last change in condition was outside of the discrepancy time.
			TRUE	Both inputs (A/B) = TRUE and last status change within the discrepancy time.
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

### Possible results for DIAGCODE:

Value [hex]	Description
0000	The function block is inactive. Start state.
8001	The function block is active, safety output = FALSE. Initialisation.
8000	The function block is active and there is no error state. Safety output = TRUE
8004	1st input = TRUE, waiting for 2nd input. Time monitoring has been started.
8014	1st input = TRUE, waiting for 2nd input. Time monitoring has been started.
8005	One channel = FALSE, waiting for the 2nd channel = FALSE. Time monitoring has been started.
C001	Error: Monitoring time has expired (FB waits for the 2nd channel)
C002	Error: Monitoring time has expired (FB waits for the 1st channel)
C003	Error: Monitoring time has expired (FB waits for both channels = inactive)

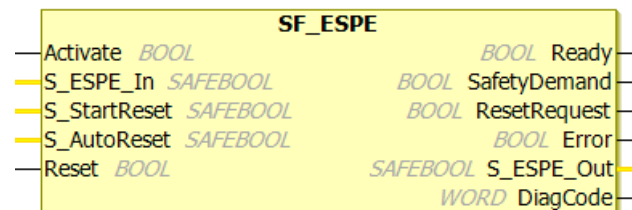
## SF\_ESPE

27329

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27123

The function block monitors electro-sensitive protective equipment (ESPE), e.g. a light grid.

The function is identical with the function of the function block SF\_EmergencyStop.

24785

The reset behaviour of the function block can be set via the inputs S\_StartReset and S\_AutoReset.



### WARNING

For **PowerOn** and **S\_StartReset = TRUE**, the function block starts operating without acknowledging the start.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.



### WARNING

When the **protective equipment is reset** and **S\_AutoReset = TRUE**, the function block starts operating without manual acknowledgement.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.

## Input parameters

27221

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_ESPE_In	SAFEBOOL	Input signal of the electro-sensitive protective equipment (ESPE)	FALSE	(initial value) ESPE has triggered.
			TRUE	ESPE not triggered.
S_StartReset	SAFEBOOL	Selection manual / automatic reset when the function block is activated.	FALSE	(initial value) When the function block has been activated, manual reset is required.
		Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	TRUE	When the function block has been activated, an automatic reset will take place.
S_AutoReset	SAFEBOOL	Selection of manual / automatic reset after the protective equipment has been reset (e.g. E-stop, ESPE, etc.).	FALSE	(initial value) After resetting the protective equipment, manual reset is required.
		Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	TRUE	After resetting the protective equipment, an automatic reset takes place.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27094

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_ESPE_Out	SAFEBOOL	Output for safety-related feedback.	FALSE	(initial value): Safety-related output is deactivated: ESPE has triggered. Reset is required, or an internal error has occurred
			TRUE	Safety-related output is active
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8001	Activation = TRUE, the FB is active. Verification whether a manual reset is required.
8002	Function block is ready: Activation = TRUE Function block waits for safety-related input=TRUE
8003	Function block is active: Activate = TRUE, safety input = TRUE FB waits for edge to reset = FALSE => TRUE
8004	Activate = TRUE Safety request detected Verify if S_AutoReset = TRUE or FALSE Function block waits for safety-related input=TRUE
8005	Activation = TRUE Safety-related input=TRUE FB waits for the rising edge to reset (FALSE => TRUE) or for S_AUTORESET=TRUE
8000	Activation = TRUE Safety-related input=TRUE Safety output = TRUE
C001	Error because permanently reset = TRUE in 8003, waiting for rest = FALSE
C002	Error because permanently reset = TRUE in 8005, waiting for rest = FALSE



## SF\_FootSwitch

27330

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27132

The function block evaluates the signals of a foot switch with three switching positions according to DIN EN 60204 section 9.2.5.8.

25160

The function block's reset behaviour can be set via the input S\_AutoReset.



### WARNING

When the **protective equipment is reset** and **S\_AutoReset = TRUE**, the function block starts operating without manual acknowledgement.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.

## Input parameters

27230

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_SafetyActive	SAFEBOOL	Indicates if the safe mode is active or not.	FALSE	Initial value Safe mode is inactive.
			TRUE	Safe mode is active.
S_FootSwitchCh1	SAFEBOOL	Signal of the contacts E1 and E2 of the connected foot switch.	FALSE	Initial value Connected switching contacts are open
			TRUE	Connected switching contacts are closed
S_FootSwitchCh2	SAFEBOOL	Signal of the contacts E3 and E4 of the connected foot switch.	FALSE	Initial value Connected switching contacts are open
			TRUE	Connected switching contacts are closed
S_AutoReset	SAFEBOOL	Selection of manual / automatic reset after the protective equipment has been reset (e.g. E-stop, ESPE, etc.).  Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	FALSE	(initial value) After resetting the protective equipment, manual reset is required.
			TRUE	After resetting the protective equipment, an automatic reset takes place.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27103

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_FootSwitchOut	SAFEBOOL	Safety-related output	FALSE	Initial value Safety-related release output is inactive
			TRUE	Safety-related release output is active
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

### Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8004	The safety-related release output is inactive (FALSE).
8005	The safety-related release output is active (TRUE).
8006	The safety-related release output is active (TRUE) and the switch is in position 1.
8810	The safety-related release output is active (TRUE) and the switch is in position 3.
8000	The safety-related release output is active (TRUE) and the switch is in position 2.
C141	Error: Reset permanently TRUE in the Operation Error 2 (C140) state.
C181	Error: Reset permanently TRUE in the Operation Error 4 (C180) state.
C120	Error: Switch not in position 1 after S_SafetyActive (safe mode) has been activated.
C140	Error: Switch in position 1 after Operation Error 1 (C120) state.
C160	Error: Switch in position 2 after position 3.
C180	Error: Switch not in position 2 after Operation Error 3 (C160) state.

## SF\_GuardLocking

27331

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27127

The SF\_GuardLocking function block controls and monitors the access to a hazardous area by means of an interlocking guard with guard locking.

24785

The reset behaviour of the function block can be set via the inputs S\_StartReset and S\_AutoReset.



### WARNING

For **PowerOn** and **S\_StartReset = TRUE**, the function block starts operating without acknowledging the start.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.



### WARNING

When the **protective equipment is reset** and **S\_AutoReset = TRUE**, the function block starts operating without manual acknowledgement.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.

## Input parameters

27225

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_GuardMonitoring	SAFEBOOL	Monitors the locked guard	FALSE	Initial value Protective equipment open
			TRUE	Guard closed
S_SafetyActive	SAFEBOOL	Status of the dangerous zone	FALSE	Initial value Dangerous zone in non safe state
			TRUE	Dangerous zone in safe state
S_GuardLock	SAFEBOOL	Status of the mechanic guard locking	FALSE	Initial value Guard locking is not engaged
			TRUE	Guard locking is engaged
S_UnlockRequest	BOOL	User intervention. Unlocking request for the guard locking.	FALSE	Initial value No request.
			TRUE	Request sent.
S_StartReset	SAFEBOOL	Selection manual / automatic reset when the function block is activated.	FALSE	(initial value) When the function block has been activated, manual reset is required.
		Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	TRUE	When the function block has been activated, an automatic reset will take place.
S_AutoReset	SAFEBOOL	Selection of manual / automatic reset after the protective equipment has been reset (e.g. E-stop, ESPE, etc.).	FALSE	(initial value) After resetting the protective equipment, manual reset is required.
		Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	TRUE	After resetting the protective equipment, an automatic reset takes place.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27098

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_GuardLocked	SAFEBOOL	Interface to the dangerous zone	FALSE	Initial value non safe state
			TRUE	safe state: - Protective equipment is closed and locked - There is no unlocking request - Reset has been carried out
S_UnlockGuard	SAFEBOOL	Signal to unlock the protective equipment	FALSE	Initial value Lock
			TRUE	Unlock
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

## Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	Protective equipment is closed and locked.
8001	The function block is active, safety outputs = FALSE. Initialisation.
8003	Protective equipment is closed and locked. Waiting for reset.
8011	Waiting for locking
8012	Protective equipment unlocked and open.
8013	Protective equipment closed, but not locked.
8014	Return of the S_SafetyActive signal. Waiting for operator acknowledgement.
C001	Error: Reset permanently TRUE in the Init (8001) state. (Only in case of manual reset.)
C002	Error: Reset permanently TRUE in the Safety Return (8014) state.
C003	Error: Reset permanently TRUE in the Wait for Reset (8003) state.
C004	Error: Safety function triggered. Electro-sensitive equipment open or not locked.

## SF\_GuardMonitoring

27332

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27124

The function block SF\_GuardMonitoring is used to monitor protective equipment with a 2-channel safety switch.

24785

The reset behaviour of the function block can be set via the inputs S\_StartReset and S\_AutoReset.



### WARNING

For **PowerOn** and **S\_StartReset = TRUE**, the function block starts operating without acknowledging the start.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.



### WARNING

When the **protective equipment is reset** and **S\_AutoReset = TRUE**, the function block starts operating without manual acknowledgement.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.



## Input parameters

27222

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_GuardSwitch1	SAFEBOOL	Fail-safe sensors 1	FALSE	Initial value Switching contact is open
			TRUE	Switching contact is closed.
S_GuardSwitch2	SAFEBOOL	Fail-safe sensors 2	FALSE	Initial value Switching contact is open
			TRUE	Switching contact is closed
DiscrepancyTime	TIME	Maximum monitoring time for the discrepancy check of both input channels.	permissible: T#0ms... T#49,71d	minimum time value maximum time value
S_StartReset	SAFEBOOL	Selection manual / automatic reset when the function block is activated.  Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	FALSE	(initial value) When the function block has been activated, manual reset is required.
			TRUE	When the function block has been activated, an automatic reset will take place.
S_AutoReset	SAFEBOOL	Selection of manual / automatic reset after the protective equipment has been reset (e.g. E-stop, ESPE, etc.).  Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	FALSE	(initial value) After resetting the protective equipment, manual reset is required.
			TRUE	After resetting the protective equipment, an automatic reset takes place.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27095

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_GuardMonitoring	SAFEBOOL	Condition of the input signals S_GuardSwitch1 and S_GuardSwitch2	FALSE	Initial value One of the two inputs S_GuardSwitchX = FALSE.
			TRUE	Both inputs S_GuardSwitch = TRUE. There is no error and the change of both inputs to TRUE within the discrepancy time has been confirmed by Reset.
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	The function block is active and there is no error state. Safety output = TRUE
8001	The function block is active, safety output = FALSE. Initialisation.
8002	Full switching sequence required.
8003	Function block is ready: both safety inputs = TRUE FB waits for edge to reset = FALSE => TRUE
8012	Both safety inputs = FALSE
8004	1st input = TRUE, waiting for 2nd input. Time monitoring has been started.
8014	1st input = TRUE, waiting for 2nd input. Time monitoring has been started.
8005	Protective equipment active. Safety inputs = TRUE FB verifies for S_AutoReset = TRUE
C001	Error: reset permanently TRUE
C011	Maximum monitoring time exceeded while waiting for S_GuardSwitch2 = TRUE
C012	Maximum monitoring time exceeded while waiting for S_GuardSwitch1 = TRUE

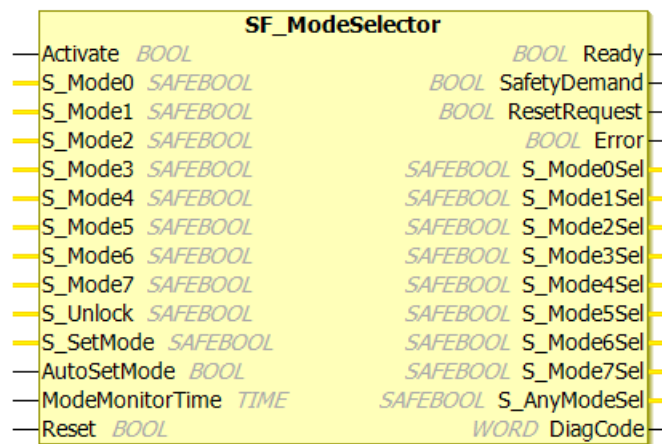
## SF\_ModeSelector

27336

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27121

This function block enables reliable switching between up to 8 operating modes of a machine or installation.

**Procedure:**

1. unlock current operating mode
2. select new operating mode
3. lock selected operating mode (deselectable)

The function block monitors the process and duration of the switching operation.

## Input parameters

27219

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_ModeX	SAFEBOOL	Input X; connect with the switch output X (with X = 0..7).	FALSE	Operating mode x was not requested.
			TRUE	Operating mode X is requested.
S_Unlock	SAFEBOOL	Unlocks the current operating mode X (with X = 0..7).	FALSE	The active output S_ModeXSel is locked. Changes made to any of the inputs S_ModeX do not result in changes to the outputs, not even in the event of a rising edge at the input S_SetMode. (with X = 0..7)
			TRUE	The currently active output S_MODExSEL (x=0...7) is unlocked, changes are possible
S_SetMode	SAFEBOOL	Confirm changed operating mode	FALSE	Each change made on an input S_ModeX after TRUE leads to: S_AnyModeSel/S_ModeXSel = FALSE.
			FALSE ⇒ TRUE (edge)	The output S_ModeXSel corresponding to the selected output S_ModeX becomes true TRUE (with X = 0..7)
AutoSetMode	BOOL	Configuration of the switchover confirmation.	FALSE	(initial value): A changed operating mode must be confirmed with a rising edge on the S_Set_Mode input. Only then, the corresponding output will become S_ModeXSel=TRUE.
			TRUE	A valid switchover from one S_MODEx input to another one directly leads to a change of the S_MODExSEL outputs (provided S_UNLOCK=TRUE).
ModeMonitorTime	Time	Maximum permissible time for a change on the inputs S_ModeX (with X = 0..7).	T#0ms	Initial value
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

27092

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_ModeXSel	SAFEBOOL	Indicates that the operating mode X has been selected and confirmed (with X = 0..7).	FALSE	Mode X has not been selected or is not active.
			TRUE	Mode X has been selected and is active.
S_AnyModeSel	SAFEBOOL	Indicates that one of the 8 operating types has been selected and confirmed.	FALSE	Initial value No operating type has been selected with S_ModeX.
			TRUE	One of the 8 operating modes is selected and confirmed.
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

### Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8005	State after activation or if S_ModeX has changed or after an error state has been reset.
8000	Valid selection of the operating type, not yet locked.
8004	Selection of the operating type is locked.
C001	Error: Several SModeX inputs on TRUE, e.g. due to short-circuit.
C002	Error: All S_ModeX inputs longer than MonitoringTime on FALSE, e.g. open circuit.
C003	Error: Static reset signal during short-circuit detected.
C004	Error: Static reset signal detected in an open circuit.

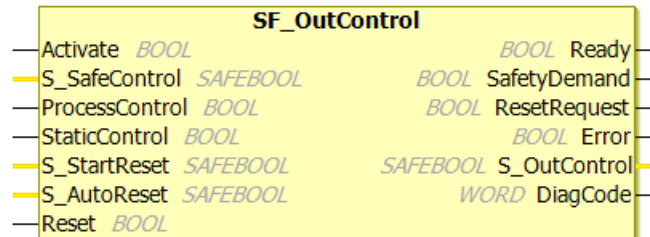
## SF\_OutControl

27337

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27130

The function block SF\_OutControl controls a safe output with a signal from the standard application (ProcessControl/BOOL) and a signal from the safety application (S\_SafeControl/SAFEBOOL).

24785

The reset behaviour of the function block can be set via the inputs S\_StartReset and S\_AutoReset.



### WARNING

For **PowerOn** and **S\_StartReset = TRUE**, the function block starts operating without acknowledging the start.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.



### WARNING

When the **protective equipment is reset** and **S\_AutoReset = TRUE**, the function block starts operating without manual acknowledgement.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.

## Input parameters

27228

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_SafeControl	SAFEBOOL	Control signal of the preceding safe function block	FALSE	Initial value The preceding safe function block is in the safe state.
			TRUE	The safety-related output of the preceding safe function block is active (TRUE).
ProcessControl	BOOL	Control signal of the functional applications.	FALSE	Initial value Request to set output S_OutControl to FALSE.
			TRUE	Request to set output S_OutControl to TRUE.
StaticControl	BOOL	Optional condition for ProcessControl	FALSE	Initial value Edge change FALSE => TRUE is required at the ProcessControl input, after the function block has been activated or the after the safety-related release output has been deactivated.
			TRUE	No edge change is required at the ProcessControl input, after the function block has been activated or the after the safety-related release output has been deactivated.
S_StartReset	SAFEBOOL	Selection manual / automatic reset when the function block is activated.	FALSE	(initial value) When the function block has been activated, manual reset is required.
		Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	TRUE	When the function block has been activated, an automatic reset will take place.
S_AutoReset	SAFEBOOL	Selection of manual / automatic reset after the protective equipment has been reset (e.g. E-stop, ESPE, etc.).	FALSE	(initial value) After resetting the protective equipment, manual reset is required.
		Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	TRUE	After resetting the protective equipment, an automatic reset takes place.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error



## Output parameters

27101

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_OutControl	SAFEBOOL	Controls connected actuators	FALSE	Initial value Deactivate connected actuators
			TRUE	Activated connected actuators
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8001	Start disable for function block activation is active. Reset required.
8002	The safety-related output of the upstream function block is not active (FALSE).
8003	Reset required after re-occurrence of the S_SafeControl signal of the upstream safety function block.
8010	Process control is not active.
8000	The safety-related output of the upstream function block is active (TRUE).
C001	Error: Reset permanently TRUE in the Init (8001) state. (Only in case of manual reset.)
C002	Error: Input reset permanently TRUE in the Lock (8003) state. (Only in case of manual reset.)
C010	Error: ProcessControl permanently TRUE in the Output Disable (8010) state. Edge change at ProcessControl required.
C111	Error: Simultaneous rising edge at the reset and ProcessControl inputs in the Init (8001) state.
C211	Error: Simultaneous rising edge at the reset and ProcessControl inputs in the Lock (8003) state.

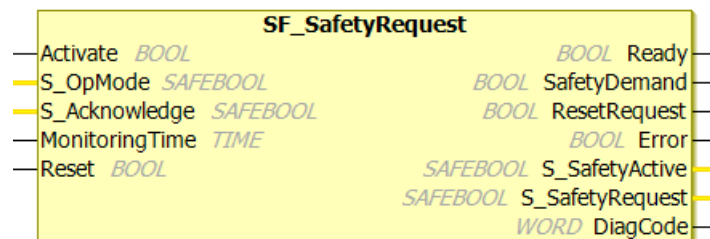
## SF\_SafetyRequest

27341

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

## Description

27129

The function block SF\_SafetyRequest offers an interface with back channel to a safety actuator, e.g. safety drive or safety valve for the following purposes:

- bringing the actuator into the safe state.
- monitoring whether the actuator has assumed the safe state within the requested time.

## Input parameters

27227

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_OpMode	SAFEBOOL	Input to request the safe state for the actuator.	FALSE	Initial value Request for the safe state of the actuator.
			TRUE	No request for the safe state of the actuator.
S_Acknowledge	SAFEBOOL	Feedback about the operating mode from the safe actuator	FALSE	Initial value Non-safe operating mode
			TRUE	Safe mode
MonitoringTime	TIME	Monitoring time maximum permissible response time between the safety request (edge S_OpMode = TRUE => FALSE) and the confirmation of the actuator (edge S_Acknowledge = FALSE => TRUE)	T#0s	Initial value
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_SafetyActive	SAFEBOOL	Status of the connected safety actuator	FALSE	Initial value Actuator is in the non safe state.
			TRUE	Actuator is in the safe state.
S_SafetyRequest	SAFEBOOL	Request to bring the actuator into the safe state.	FALSE	Initial value Safe mode has been requested
			TRUE	Safe mode has not been requested
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	The function block is active and there is no error state. Safety output = TRUE
8001	The function block is active, safety output = FALSE. Waiting for edge (FALSE => TRUE) at Reset.
8002	Function block is active. Operating status of the FB is without request of the safe state for the actuator.
8012	Function block is active: Waiting for request to set the actuator in the safe state.
8003	Waiting for confirmation that the actuator has adopted the safe state.
8005	Error eliminated. The requirements for the safe state of the actuator must be taken back once (S_OpMode = TRUE) before the function block can be used again.
C002	Error: Confirmation lost in the safe state.
C003	Error: A confirmation that the actuator has switched to the safe state during the monitoring time has not been detected.
C004	Error: Reset permanently TRUE in the Acknowledge Lost (C002) state
C005	Error: Reset permanently TRUE in the MonitoringTime Elapsed (C003) state

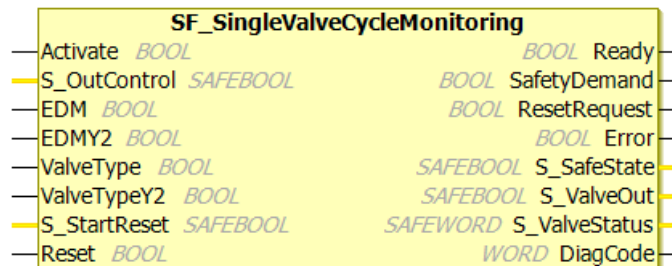
## SF\_SingleValveCycleMonitoring

27342

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### Description

27134

This function block is used to monitor a cartridge valve. Per machine cycle, a signal change of the valve feedback signal EDM and an edge at EDMY2 must be recognised. If this is not the case, the output S\_ValveOut goes in the safe state.

The safe feedback state of EDM is set via the parameter ValveType. The safe feedback state of EDMY2 is set via the parameter ValveTypeY2.

27134



The function block SF\_SingleValveCycleMonitoring changes from state 8804 with priority 1 into the state C0C0.

Information: In the PLCopen specification\*, it is not clearly described whether state CxC0 or CxD0 will be changed to.

\* → PLCopen TC5 Safety Software Technical Specification Part 4, chapter 4.4.3 "Functional Description", "State Diagram"

24885

The function block's reset behaviour can be set via the input S\_StartReset.



### WARNING

For **PowerOn** and **S\_StartReset = TRUE**, the function block starts operating without acknowledging the start.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.

## Input parameters

24070

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_OutControl	SAFEBOOL	Status of the connected valve	FALSE	Initial value Valve is in the safe state.
			TRUE	Valve is in the non safe state.
EDM	BOOL	Valve feedback	FALSE	Initial value Significance depends on the ValveType input.
			TRUE	Significance depends on the ValveType input.
EDMY2	BOOL	Valve Y2 feedback	FALSE	Initial value Significance depends on the ValveTypeY2 input.
			TRUE	Significance depends on the ValveTypeY2 input.
ValveType	BOOL	Parameter to set the polarity of the valve feedback signal for the safe state.	FALSE	Initial value The safe state of the valve must be reported back with a LOW signal to EDM. In the non safe state, there must be a HIGH signal at EDM.
			TRUE	The safe state of the valve must be reported back with a HIGH signal to EDM. In the non safe state, there must be a LOW signal at EDM.
ValveTypeY2	BOOL	Parameter to set the polarity of the feedback signal of the Y2 valve safety position.	FALSE	Initial value Safety status is reported back with a LOW signal.
			TRUE	Safety status is reported back with a HIGH signal.
S_StartReset	SAFEBOOL	Selection manual / automatic reset when the function block is activated.	FALSE	(initial value) When the function block has been activated, manual reset is required.
		Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	TRUE	When the function block has been activated, an automatic reset will take place.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameter

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_SafeState	SAFEBOOL	Safe state of the valve / valves	FALSE	Initial value The valve is not in the safe state or an error has occurred.
			TRUE	The valve is in the safe state and no error has occurred.
S_ValveOut	SAFEBOOL	Safety-related output to control the pilot valve.	FALSE	Initial value Solenoid of the pilot valve is disconnected from power
			TRUE	Solenoid of the pilot valve is connected to power
S_ValveStatus	SAFEWORD	Status signal of the safety valve This output can be used for connection with the function block SF_ValveGroupControl.	16#0000	Initial value
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	



## Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
83FF	Log on to valve group.
8401	Function block activation. Reset required.
8802	Waiting for the feedback signal (EDM) to confirm the safe coil position of the valve.
8804	The feedback signal (EDM) confirms that the valve is in the safe state.
8806	Waiting for the rising edge (FALSE => TRUE) at EDMY2.
8010	Valve is no longer in the safe state.
8020	The output S_ValveOut is set to TRUE to control the valve.
C100	Error: Identical signal curve detected at S_OutControl and Reset (rising edge FALSE => TRUE in the same cycle) during manual reset. Could be a programming error.
C001	Error: Reset permanently TRUE in the Init (8401) state during manual reset.
C011	Error: Reset permanently TRUE in the EDM Error X10 (CX10) state.
C041	Error: Reset permanently TRUE in the EDM Error X40 (CX40) state.
C0C1	Error: Reset permanently TRUE in the EDM Error XC0 (CXC0) state.
C0D1	Error: Reset permanently TRUE in the EDM Error XD0 (CXD0) state.
Cx10	Feedback error; the EDM1 signal is invalid in the 8802 state. EDM1 = FALSE, when activating the safety function.
Cx40	The signal on EDM is invalid in the 8020 state. In the 8020 state, the EDM signal is active if EDMY2 becomes inactive, i.e. a new cycle is detected.
CxC0	The signal at EDMY2 becomes inactive in the 8010 state. A new cycle is detected before the valve is switched off.
CxD0	S_OutControl becomes TRUE in the 8806 state. EDMY2 stays inactive, i.e. the cycle of the valve Y2 is not finished.

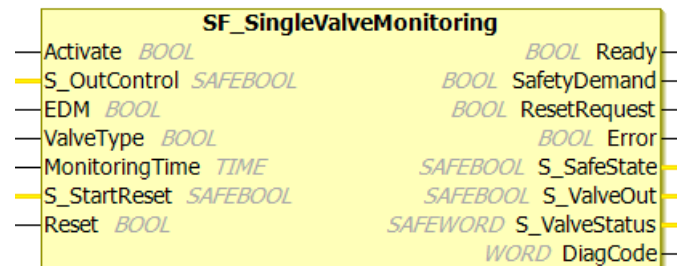
## SF\_SingleValveMonitoring

27343

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### Description

27133

The function block monitors the switching behaviour of fluidic valves.

A static feedback signal of the valve (monitoring of the coil position) monitors the safety condition of the valve.

The value in MonitoringTime is used to monitor the feedback signal.

24885

The function block's reset behaviour can be set via the input S\_StartReset.



### WARNING

For **PowerOn** and **S\_StartReset = TRUE**, the function block starts operating without acknowledging the start.

- > Dangerous restart possible.
- > Risk of personal injuries and/or damage to property.
- ▶ Ensure that there is no danger for persons or the machine/installation during restart.
- ▶ Implement a separate restart protection.

## Input parameters

27231

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_OutControl	SAFEBOOL	Status of the connected valve	FALSE	Initial value Valve is in the safe state.
			TRUE	Valve is in the non safe state.
EDM	BOOL	Valve feedback	FALSE	Initial value Significance depends on the ValveType input.
			TRUE	Significance depends on the ValveType input.
ValveType	BOOL	Parameter to set the polarity of the valve feedback signal for the safe state.	FALSE	Initial value The safe state of the valve must be reported back with a LOW signal to EDM. In the non safe state, there must be a HIGH signal at EDM.
			TRUE	The safe state of the valve must be reported back with a HIGH signal to EDM. In the non safe state, there must be a LOW signal at EDM.
MonitoringTime	TIME	Monitoring time in [ms] Within this time interval, the valve must have reported via the EDM input that it is no longer in the safe state.	T#0ms	Initial value
S_StartReset	SAFEBOOL	Selection manual / automatic reset when the function block is activated.  Detailed information about the operating principle: → PLCopen Safety specifications: <a href="http://www.plcopen.org">www.plcopen.org</a> > PLCopen Safety	FALSE	(initial value) When the function block has been activated, manual reset is required.
			TRUE	When the function block has been activated, an automatic reset will take place.
Reset	BOOL	Reset the function block after an error.	FALSE	Initial value
			TRUE	Resetting the error

## Output parameters

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_SafeState	SAFEBOOL	Safe state of the valve / valves	FALSE	Initial value The valve is not in the safe state or an error has occurred.
			TRUE	The valve is in the safe state and no error has occurred.
S_ValveOut	SAFEBOOL	Safety-related output to control the safety valve.	FALSE	Initial value Solenoid of the valve is disconnected from power
			TRUE	Solenoid of the valve is connected to power
S_ValveStatus	SAFEWORD	Status signal of the safety valve This output can be used for connection with the function block SF_ValveGroupControl.	16#0000	Initial value
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

## Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
83FF	Log on to valve group.
8401	Function block activation. Reset required.
8802	Waiting for feedback signal (EDM) to confirm the safe coil position of the valve. Monitoring time starts when entering the state.
8804	The feedback signal (EDM) confirms that the valve is in the safe state.
8010	Valve is no longer in the safe state.
8020	Waiting for feedback signal (EDM) to confirm that the valve is no longer in the safe state.
C100	Error: Identical signal curve detected at S_OutControl and Reset (rising edge FALSE => TRUE in the same cycle) during manual reset. Could be a programming error.
C001	Error: Reset permanently TRUE in the Init (8401) state during manual reset.
C011	Error: Reset permanently TRUE in the EDM Error X10 (CX10) state.
C041	Error: Reset permanently TRUE in the EDM Error X40 (CX40) state.
C071	Error: Reset permanently TRUE in the EDM Error x70 (Cx70) state.
Cx10	Error: Non safe valve state requested while waiting for the safe state.
Cx40	Error: The signal at EDM is invalid in state 8802 or 8804. In state 8802, it was not reported back to EDM during the monitoring time that the valve went into the safe state. In state 8804, it was reported to EDM that the valve went into the non safe state even though the safe state had been requested.
Cx70	Error: The signal at EDM is invalid in state 8010 or 8020. The safe state of the valve has been reported to EDM even though the non safe state is requested.

## SF\_TwoHandControlTypeII

27344

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### Description

27125

The FB SF\_TwoHandControlTypeII is used to implement a two-hand control according to EN 574, section 4 type II.

If safety switch 1 (S\_Button1) and safety switch 2 (S\_Button2) are set to TRUE, the safety-related output S\_TwoHandOut is set to TRUE. The function block also monitors whether both buttons are released before the output S\_TwoHandOut is set to TRUE again.

### Input parameters

27223

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_Button1	SAFEBOOL	Safety input / input channel for safety switch 1 (for EN 954-1 category 3 or 4: 2 complementary contacts)	FALSE	Initial value Safety switch 1 released, contact open
			TRUE	Safety switch 1 engaged, contact closed
S_Button2	SAFEBOOL	Safety input / input channel for safety switch 2 (for EN 954-1 category 3 or 4: 2 complementary contacts)	FALSE	Initial value Safety switch 2 released, contact open
			TRUE	Safety switch 2 engaged, contact closed

## Output parameters

27096

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_TwoHandOut	SAFEBOOL	Safety-related output	FALSE	Initial value No correct two-hand control carried out.
			TRUE	Safety switch 1 and safety switch 2 engaged and no error has occurred. Correct two-hand control carried out.
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

## Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	Both safety inputs are active (TRUE). Safety output = TRUE
8001	The function block is active, safety output = FALSE. Initialisation.
8004	Both safety inputs are not active (FALSE). Safety output = FALSE
8005	Safety input 1 is active (TRUE). Safety output = FALSE
8006	Safety input 2 is active (TRUE). Safety output = FALSE
8007	Safety input 2 is inactive (FALSE), safety output was TRUE and is now FALSE. Safety input 1 is active (TRUE).
8008	Safety input 1 is inactive (FALSE), safety output was TRUE and is now FALSE. Safety input 2 is active (TRUE).
8009	Safety output was TRUE and is now FALSE. Safety-related input 1 is still active or active again. Safety-related input 2 is still active or active again. Function block waits for both safety inputs=FALSE.
8019	Incorrect activation of the safety inputs. Function block waits for both safety inputs=FALSE.
C001	Error: Safety input 1 was active (TRUE) during activation of the function block.
C002	Error: Safety input 2 was active (TRUE) during activation of the function block.
C003	Error: Safety input 1 and safety input 2 were active (TRUE) during activation of the function block.



## SF\_TwoHandControlTypeIII

27345

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### Description

27126

The FB SF\_TwoHandControlTypeIII is used to implement two-hand control with a fixed monitoring time of 500ms according to EN 574, section 4 type III.

If safety switch 1 (S\_Button1) and safety switch 2 (S\_Button2) are engaged within 500 ms and set (to TRUE), the safety-related output S\_TwoHandOut will be set to TRUE. The function block also monitors whether both buttons are released before the output S\_TwoHandOut is set again to TRUE.

### Input parameters

27224

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_Button1	SAFEBOOL	Safety input / input channel for safety switch 1 (for EN 954-1 category 3 or 4: 2 complementary contacts)	FALSE	Initial value Safety switch 1 released, contact open
			TRUE	Safety switch 1 engaged, contact closed
S_Button2	SAFEBOOL	Safety input / input channel for safety switch 2 (for EN 954-1 category 3 or 4: 2 complementary contacts)	FALSE	Initial value Safety switch 2 released, contact open
			TRUE	Safety switch 2 engaged, contact closed

## Output parameters

24039

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_TwoHandOut	SAFEBOOL	Safety-related output	FALSE	Initial value No correct two-hand control carried out.
			TRUE	Safety switch 1 and safety switch 2 have been engaged within 500ms and no error has occurred. Correct two-hand control carried out.
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	

## Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state.
8000	Both safety inputs are active (TRUE). Safety output = TRUE
8001	The function block is active, safety output = FALSE. Initialisation.
8004	Both safety inputs are not active (FALSE). Safety output = FALSE
8005	Safety switch 1 engaged. Beginning of the monitoring time. Safety output = FALSE
8006	Safety switch 2 engaged. Beginning of the monitoring time. Safety output = FALSE
8007	Safety input 2 is inactive (FALSE), safety output was TRUE and is now FALSE. Safety input 1 is active (TRUE).
8008	Safety input 1 is inactive (FALSE), safety output was TRUE and is now FALSE. Safety input 2 is active (TRUE).
8009	Safety output was TRUE and is now FALSE. Safety-related input 1 is still active or active again. Safety-related input 2 is still active or active again. Function block waits for both safety inputs=FALSE.
8019	Incorrect activation of the safety inputs. Function block waits for both safety inputs=FALSE.
C001	Error: Safety input 1 was active (TRUE) during activation of the function block.
C002	Error: Safety input 2 was active (TRUE) during activation of the function block.
C003	Error: Safety input 1 and safety input 2 were active (TRUE) during activation of the function block.
C004	Error: Safety switch 1 was not engaged and safety switch 2 was engaged after the monitoring time = 500 ms was over.
C005	Error: Safety switch 2 was not engaged and safety switch 1 was engaged after the monitoring time = 500 ms was over.
C006	Error: Safety switch 1 was engaged and safety switch 2 was engaged after the monitoring time = 500 ms was over.

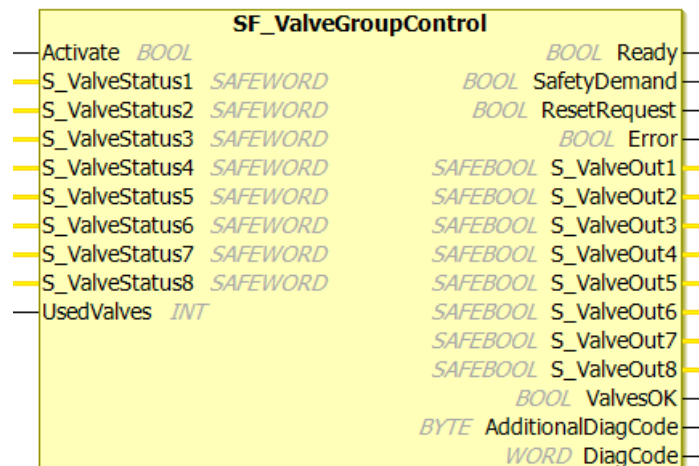
## SF\_ValveGroupControl

27346

**Function block type:** Safety function block (SF)

**Library:** ifmPLCopenSafe.library

**Symbol in CODESYS:**



Observe the information concerning the handling of the function block in standard / safety PLC and standard / safety PRG: → **Using the function blocks** (→ p. [207](#))

### Description

27136

This function block combines the connected safe function blocks to control safety valves (**SF\_DoubleValveMonitoring** (→ p. [400](#)) or **SF\_SingleValveMonitoring** (→ p. [446](#)) or **SF\_SingleValveCycleMonitoring** (→ p. [442](#))) in one group.

If there is an error on one S\_ValveStatusX input (configured via UsedValves), all safety-related outputs will go to S\_ValveOutX. These signals can, for example, be used to switch off the valves of the group at once.

Ass outputs S\_ValveOutX stay FALSE as long as there is an error in one of the configured safety-related inputs S\_ValveStatusX.

27136



Deviation of the function block SF\_ValveGroupControl1 from the PLCopen specification\*:

Instead of status 8000, the statuses 8020 and 8010 of the connected safety valve function blocks (at the inputs S\_ValveStatusX) are interpreted as ValvesOK.

All safe non error states as well as several transition states are evaluated as ValvesOK.

\* → PLCopen TC5 Safety Software Technical Specification Part 4, chapter 4.6.3 "Functional Description", paragraph 2

## Input parameters

27234

Parameter	Data type	Description	Possible values	
Activate	BOOL	Activation of the function block	FALSE	The function block is not being executed. (default)
			TRUE	The function block is being executed.
S_ValveStatusX	SAFEWORD	Status signal of the corresponding safety valve function block, e.g. SF_SingleValveMonitoring. With X = 1..8	16#0000	Initial value
UsedValves	INT	Number of safety valves combined in the group	0	Initial value
			0..8	Possible values

## Output parameters

27107

Parameter	Data type	Description	Possible values	
Ready	BOOL	Indicates that the function block is active and that the output signal is valid.	FALSE	inactive, function block is not being executed.
			TRUE	active
S_ValveOutX	SAFEBOOL	Safety-related output to control the safety valve connected via a safety function block With X = 1..8	FALSE	Initial value Deactivate a valve connected via safety function block
			TRUE	Activate a valve connected via safety function block
ValvesOK	BOOL	Combined status signal for all connected safety valves	FALSE	Initial value At least one connected valve is not in the safe state or an error has occurred
			TRUE	All connected valves are in the safe state, no error has occurred.
SafetyDemand	BOOL	Indicates if the function block has responded to a safety request.	FALSE	Initial value No safety request. The safety-related release output of the function block is active (TRUE).
			TRUE	The function block has responded to a safety request. The safety-related release output of the function block is inactive (FALSE).
ResetRequest	BOOL	Reset request	FALSE	Reset not requested
			TRUE	Reset requested
Error	BOOL	Fail condition	FALSE	No error has occurred
			TRUE	one error has occurred, and the function block is in the error state.
DiagCode	WORD	Diagnostic code. Information about the current function block status.	→ List below.	
AdditionalDiagCode	BYTE	Additional diagnostic code. Indicates which valve is in the error state, has logged off via 0000 at S_ValveStatusX or is in the non-safe state during the ValvesOK (8000) state (valve function block state 8010 / 8020). Each valve is represented by a bit.	16#00	Initial value  → List below.

## Possible results for DIAGCODE:

Value [hex]	Description
0000	The function bloc is inactive. Start state. AdditionalDiagCode = 00
8001	The function block is active, safety outputs = FALSE. Initialisation. AdditionalDiagCode = 00
8002	Function block is ready. Waiting for the login sequence of the valves connected via safety function blocks. AdditionalDiagCode = XX (depends on which connected valve function block is to be used but which is not yet logged on.)
8006	All valves connected via safety function blocks are logged on. AdditionalDiagCode = 00
80X4	Valve X has logged off. With X = 1..8. 8014 = valve 1 has logged off. AdditionalDiagCode = 01 8024 = valve 2 has logged off. AdditionalDiagCode = 02 8034 = valve 3 has logged off. AdditionalDiagCode = 04 8044 = valve 4 has logged off. AdditionalDiagCode = 08 8054 = valve 5 has logged off. AdditionalDiagCode = 10 8064 = valve 6 has logged off. AdditionalDiagCode = 20 8074 = valve 7 has logged off. AdditionalDiagCode = 40 8084 = valve 8 has logged off. AdditionalDiagCode = 80
8000	The connected valves are OK. The outputs S_ValveOut1 .. S_ValveOut8 are either switched on or off, depending on the corresponding input status S_ValveStatus1 .. S_ValveStatus8. If S_ValveStatusX = 16#10 or = 16#20, then S_ValveOutX = TRUE, otherwise S_ValveOutX = FALSE. AdditionalDiagCode = XX (depends on which connected valve function block has status 8000.)
C000	Error: Parameter UsedValves outside of the value range. AdditionalDiagCode = 00
C001	Error: Reset permanently TRUE in status 8001 AdditionalDiagCode = 00
C010	Error: Reset permanently TRUE in status C000 AdditionalDiagCode = 00
C020	Error: Reset permanently TRUE in status 80X4 AdditionalDiagCode = 00
C4X0	Error: Error detected on valve X. With X = 1..8. C410 = error detected on valve 1. AdditionalDiagCode = 01 C420 = error detected on valve 2. AdditionalDiagCode = 02 C430 = error detected on valve 3. AdditionalDiagCode = 04 C440 = error detected on valve 4. AdditionalDiagCode = 08 C450 = error detected on valve 5. AdditionalDiagCode = 10 C460 = error detected on valve 6. AdditionalDiagCode = 20 C470 = error detected on valve 7. AdditionalDiagCode = 40 C480 = error detected on valve 8. AdditionalDiagCode = 80

Value [hex]	Description
C5X0	Error: Invalid login sequence at valve X. Valve X is adjusted as used, but not logged on. With X = 1..8. C510 = Invalid login sequence of valve 1. AdditionalDiagCode = 01 C520 = invalid login sequence of valve 2. AdditionalDiagCode = 02 C530 = invalid login sequence of valve 3. AdditionalDiagCode = 04 C540 = invalid login sequence of valve 4. AdditionalDiagCode = 08 C550 = invalid login sequence of valve 5. AdditionalDiagCode = 10 C560 = invalid login sequence of valve 6. AdditionalDiagCode = 20 C570 = invalid login sequence of valve 7. AdditionalDiagCode = 40 C580 = invalid login sequence of valve 8. AdditionalDiagCode = 80



# 13     Troubleshooting

<b>Content</b>	
Error classes.....	462
Error messages .....	462
Messages / diagnostic codes of the function blocks .....	463

39632

## 13.1 Error classes

39518

An error is classified according to its possible impact. The error class determines how the system reacts when a specific error occurs.

Error class		Description	Reaction
<b>A</b>	<b>Fatal Error</b>	The overall integrity of the controller is no longer guaranteed. Errors in the central components of the controller that affect the behaviour of other components.	<ul style="list-style-type: none"> <li>The controller deactivates components allocated to the PLC.</li> <li>The controller deactivates the concerned PLC.</li> <li>The controller saves the information in the error log.</li> </ul>
<b>B</b>	<b>Serious error</b>	One or several PLCs can no longer be executed.	<ul style="list-style-type: none"> <li>The controller deactivates components allocated to the PLC.</li> <li>The controller deactivates the concerned PLC.</li> <li>The controller saves the information in the error log.</li> </ul>
<b>C</b>	<b>Component error</b>	Error in a controller component; The function of one or several components of the controller is no longer guaranteed.	<ul style="list-style-type: none"> <li>The controller puts the affected function into a defined state.</li> <li>The controller reports the error to the application.</li> <li>The controller saves the information in the error log.</li> </ul>
<b>D</b>	<b>Periphery error</b>	Errors on or in the periphery; a function can no longer be executed.	<ul style="list-style-type: none"> <li>The controller puts the affected function into a defined state.</li> <li>The controller reports the error to the application.</li> <li>The controller saves the information in the error log.</li> <li>The error can be reset in the application (as often as required).</li> </ul>

## 13.2 Error messages

39517

(Most) FBs provide, among others, the following signals at their outputs.

► Evaluate these signals in the application!

Parameters	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>Error occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	



Lists of diagnostic codes are part of the function block descriptions → **ifm function libraries** (→ p. [207](#))

### 13.3 Messages / diagnostic codes of the function blocks

39645

Status/diagnostic/error messages of the function blocs are defined in the global Enum DIAG\_INFO. They have one of the following prefixes depending on the type of message:

Prefix	Type of message	Description
STAT	Status message	Status messages contain information about the condition of the function block during the normal procedure.
DIAG	Diagnostic message	Diagnostic messages contain information about a failure event. They reset themselves after the failure event has disappeared and can optionally be evaluated by the application.
ERR	Error message	Error messages contain information about a failure event. They must be reset in the application after the failure event has disappeared.

Examples for messages / diagnostic codes:

- STAT\_INACTIVE
- DIAG\_OPEN\_CIRCUIT
- ERR\_OVERVOLTAGE



Lists of diagnostic codes are part of the function block descriptions → **ifm function libraries**  
(→ p. [207](#))

## 14 Appendix

### Content

List of inputs .....	465
List of outputs .....	471
Filter .....	476
Using the operators .....	478
Behaviour in case of floating point operations .....	480
Behaviour in case of integer operations .....	483
Mapping table [H2] user manual / ifm ecomatController CR7xxS .....	483
Directory structure and file overview .....	485
Overview of user rights .....	487
Task configuration example.....	489
ifm behaviour models for function blocks .....	497

39566

## 14.1 List of inputs

### 14.1.1 List of the inputs CR710S

24980

IEC identifier	Input type
IN0000	IN Frequency-B
IN0001	IN Frequency-B
IN0002	IN Frequency-B
IN0003	IN Frequency-B
IN0100	IN Multifunction-A
IN0101	IN Multifunction-A
IN0102	IN Multifunction-A
IN0103	IN Multifunction-A
IN0400	IN Resistor-A
IN0401	IN Resistor-A
IN0500	IN Frequency-B
IN0501	IN Frequency-B
IN0502	IN Frequency-B
IN0503	IN Frequency-B
IN0600	IN Multifunction-A
IN0601	IN Multifunction-A
IN0602	IN Multifunction-A
IN0603	IN Multifunction-A
IN0900	IN Resistor-A
IN0901	IN Resistor-A

### 14.1.2 List of the inputs CR711S

24983

IEC identifier	Input type
IN0000	IN Frequency-B
IN0001	IN Frequency-B
IN0002	IN Frequency-B
IN0003	IN Frequency-B
IN0100	IN Multifunction-A
IN0101	IN Multifunction-A
IN0102	IN Multifunction-A
IN0103	IN Multifunction-A
IN0200	IN Multifunction-A
IN0201	IN Multifunction-A
IN0202	IN Multifunction-A

IEC identifier	Input type
IN0203	IN Multifunction-A
IN0300	IN Digital-B
IN0301	IN Digital-B
IN0400	IN Resistor-A
IN0401	IN Resistor-A
IN0500	IN Frequency-B
IN0501	IN Frequency-B
IN0502	IN Frequency-B
IN0503	IN Frequency-B
IN0600	IN Multifunction-A
IN0601	IN Multifunction-A
IN0602	IN Multifunction-A
IN0603	IN Multifunction-A
IN0700	IN Multifunction-A
IN0701	IN Multifunction-A
IN0702	IN Multifunction-A
IN0703	IN Multifunction-A
IN0800	IN Digital-B
IN0801	IN Digital-B
IN0900	IN Resistor-A
IN0901	IN Resistor-A

### 14.1.3 List of the inputs CR720S

24986

IEC identifier	Input type
IN0000	IN Frequency-B
IN0001	IN Frequency-B
IN0002	IN Frequency-B
IN0003	IN Frequency-B
IN0100	IN Multifunction-A
IN0101	IN Multifunction-A
IN0102	IN Multifunction-A
IN0103	IN Multifunction-A
IN0200	IN Multifunction-A
IN0201	IN Multifunction-A
IN0202	IN Multifunction-A
IN0203	IN Multifunction-A
IN0300	IN Digital-B
IN0301	IN Digital-B

IEC identifier	Input type
IN0400	IN Resistor-A
IN0401	IN Resistor-A
IN0500	IN Frequency-B
IN0501	IN Frequency-B
IN0502	IN Frequency-B
IN0503	IN Frequency-B
IN0600	IN Multifunction-A
IN0601	IN Multifunction-A
IN0602	IN Multifunction-A
IN0603	IN Multifunction-A
IN0700	IN Multifunction-A
IN0701	IN Multifunction-A
IN0702	IN Multifunction-A
IN0703	IN Multifunction-A
IN0800	IN Digital-B
IN0801	IN Digital-B
IN0900	IN Resistor-A
IN0901	IN Resistor-A
IN1000	IN Frequency-B
IN1001	IN Frequency-B
IN1002	IN Frequency-B
IN1003	IN Frequency-B
IN1100	IN Multifunction-A
IN1101	IN Multifunction-A
IN1102	IN Multifunction-A
IN1103	IN Multifunction-A
IN1200	IN Digital-A
IN1201	IN Digital-A
IN1202	IN Digital-A
IN1203	IN Digital-A
IN1300	IN Digital-B
IN1301	IN Digital-B
IN1302	IN Digital-B
IN1303	IN Digital-B
IN1500	IN Frequency-B
IN1501	IN Frequency-B
IN1502	IN Frequency-B
IN1503	IN Frequency-B
IN1600	IN Multifunction-A

IEC identifier	Input type
IN1601	IN Multifunction-A
IN1602	IN Multifunction-A
IN1603	IN Multifunction-A
IN1700	IN Digital-A
IN1701	IN Digital-A
IN1702	IN Digital-A
IN1703	IN Digital-A



## 14.1.4 List of the inputs CR721S

24989

IEC identifier	Input type
IN0000	IN Frequency-B
IN0001	IN Frequency-B
IN0002	IN Frequency-B
IN0003	IN Frequency-B
IN0100	IN Multifunction-A
IN0101	IN Multifunction-A
IN0102	IN Multifunction-A
IN0103	IN Multifunction-A
IN0200	IN Multifunction-A
IN0201	IN Multifunction-A
IN0202	IN Multifunction-A
IN0203	IN Multifunction-A
IN0300	IN Digital-B 3.2k
IN0301	IN Digital-B 3.2k
IN0400	IN Resistor-A
IN0401	IN Resistor-A
IN0500	IN Frequency-B
IN0501	IN Frequency-B
IN0502	IN Frequency-B
IN0503	IN Frequency-B
IN0600	IN Multifunction-A
IN0601	IN Multifunction-A
IN0602	IN Multifunction-A
IN0603	IN Multifunction-A
IN0700	IN Multifunction-A
IN0701	IN Multifunction-A
IN0702	IN Multifunction-A
IN0703	IN Multifunction-A
IN0800	IN Digital-B 3.2k
IN0801	IN Digital-B 3.2k
IN0900	IN Resistor-A
IN0901	IN Resistor-A
IN1000	IN Frequency-B
IN1001	IN Frequency-B
IN1002	IN Frequency-B
IN1003	IN Frequency-B
IN1100	IN Multifunction-A

IEC identifier	Input type
IN1101	IN Multifunction-A
IN1102	IN Multifunction-A
IN1103	IN Multifunction-A
IN1200	IN Digital-A
IN1201	IN Digital-A
IN1202	IN Digital-A
IN1203	IN Digital-A
IN1300	IN Digital-B
IN1301	IN Digital-B
IN1302	IN Digital-B
IN1303	IN Digital-B
IN1400	IN Digital-A
IN1401	IN Digital-A
IN1402	IN Digital-A
IN1403	IN Digital-A
IN1500	IN Frequency-B
IN1501	IN Frequency-B
IN1502	IN Frequency-B
IN1503	IN Frequency-B
IN1600	IN Multifunction-A
IN1601	IN Multifunction-A
IN1602	IN Multifunction-A
IN1603	IN Multifunction-A
IN1700	IN Digital-A
IN1701	IN Digital-A
IN1702	IN Digital-A
IN1703	IN Digital-A
IN1800	IN Digital-A
IN1801	IN Digital-A
IN1802	IN Digital-A
IN1803	IN Digital-A

## 14.2 List of outputs

### 14.2.1 List of the outputs CR710S

39783

IEC identifier	Output type
OUT0000	OUT PWM-25-A
OUT0001	OUT PWM-25-B
OUT0002	OUT PWM-25-A
OUT0003	OUT PWM-25-B
OUT0004	OUT PWM-25-A
OUT0005	OUT PWM-25-B
OUT0006	OUT PWM-40-Bridge-A
OUT0007	OUT PWM-40-Bridge-A
OUT0100	OUT PWM-25-A
OUT0101	OUT PWM-25-B
OUT0102	OUT PWM-25-A
OUT0103	OUT PWM-25-B
OUT0104	OUT PWM-25-A
OUT0105	OUT PWM-25-B
OUT0106	OUT PWM-40-Bridge-A
OUT0107	OUT PWM-40-Bridge-A
OUT3000	OUT Supply-A
OUT3001	OUT Voltage-A

### 14.2.2 List of the outputs CR711S

39776

IEC identifier	Output type
OUT0000	OUT PWM-25-A
OUT0001	OUT PWM-25-B
OUT0002	OUT PWM-25-A
OUT0003	OUT PWM-25-B
OUT0004	OUT PWM-25-A
OUT0005	OUT PWM-25-B
OUT0006	OUT PWM-40-Bridge-A
OUT0007	OUT PWM-40-Bridge-A
OUT0008	OUT PWM-40-A
OUT0100	OUT PWM-25-A
OUT0101	OUT PWM-25-B
OUT0102	OUT PWM-25-A
OUT0103	OUT PWM-25-B
OUT0104	OUT PWM-25-A

IEC identifier	Output type
OUT0105	OUT PWM-25-B
OUT0106	OUT PWM-40-Bridge-A
OUT0107	OUT PWM-40-Bridge-A
OUT0108	OUT PWM-40-A
OUT0200	OUT PWM-25-A
OUT0201	OUT PWM-25-B
OUT0202	OUT PWM-25-A
OUT0203	OUT PWM-25-B
OUT0204	OUT PWM-25-A
OUT0205	OUT PWM-25-B
OUT0206	OUT PWM-40-Bridge-A
OUT0207	OUT PWM-40-Bridge-A
OUT0208	OUT PWM-40-A
OUT3000	OUT Supply-A
OUT3001	OUT Voltage-A

### 14.2.3 List of outputs CR720S

39746

IEC identifier	Output type
OUT0000	OUT PWM-25-A
OUT0001	OUT PWM-25-B
OUT0002	OUT PWM-25-A
OUT0003	OUT PWM-25-B
OUT0004	OUT PWM-25-A
OUT0005	OUT PWM-25-B
OUT0006	OUT PWM-40-Bridge-A
OUT0007	OUT PWM-40-Bridge-A
OUT0008	OUT PWM-40-A
OUT0100	OUT PWM-25-A
OUT0101	OUT PWM-25-B
OUT0102	OUT PWM-25-A
OUT0103	OUT PWM-25-B
OUT0104	OUT PWM-25-A
OUT0105	OUT PWM-25-B
OUT0106	OUT PWM-40-Bridge-A
OUT0107	OUT PWM-40-Bridge-A

IEC identifier	Output type
OUT0108	OUT PWM-40-A
OUT0200	OUT PWM-25-A
OUT0201	OUT PWM-25-B
OUT0202	OUT PWM-25-A
OUT0203	OUT PWM-25-B
OUT0204	OUT PWM-25-A
OUT0205	OUT PWM-25-B
OUT0206	OUT PWM-40-Bridge-A
OUT0207	OUT PWM-40-Bridge-A
OUT0208	OUT PWM-40-A
OUT0300	OUT PWM-25-A
OUT0301	OUT PWM-25-B
OUT0302	OUT PWM-25-A
OUT0303	OUT PWM-25-B
OUT0304	OUT PWM-25-A
OUT0305	OUT PWM-25-B
OUT0306	OUT PWM-40-Bridge-A
OUT0307	OUT PWM-40-Bridge-A
OUT0308	OUT PWM-40-A
OUT3000	OUT Supply-A
OUT3001	OUT Voltage-A
OUT3002	OUT Voltage-A

## 14.2.4 List of outputs

39635

IEC identifier	Output type
OUT0000	OUT PWM-25-A
OUT0001	OUT PWM-25-B
OUT0002	OUT PWM-25-A
OUT0003	OUT PWM-25-B
OUT0004	OUT PWM-25-A
OUT0005	OUT PWM-25-B

IEC identifier	Output type
OUT0006	OUT PWM-40-Bridge-A
OUT0007	OUT PWM-40-Bridge-A
OUT0008	OUT PWM-40-A
OUT0100	OUT PWM-25-A
OUT0101	OUT PWM-25-B
OUT0102	OUT PWM-25-A
OUT0103	OUT PWM-25-B
OUT0104	OUT PWM-25-A
OUT0105	OUT PWM-25-B
OUT0106	OUT PWM-40-Bridge-A
OUT0107	OUT PWM-40-Bridge-A
OUT0108	OUT PWM-40-A
OUT0200	OUT PWM-25-A
OUT0201	OUT PWM-25-B
OUT0202	OUT PWM-25-A
OUT0203	OUT PWM-25-B
OUT0204	OUT PWM-25-A
OUT0205	OUT PWM-25-B
OUT0206	OUT PWM-40-Bridge-A
OUT0207	OUT PWM-40-Bridge-A
OUT0208	OUT PWM-40-A
OUT0300	OUT PWM-25-A
OUT0301	OUT PWM-25-B
OUT0302	OUT PWM-25-A
OUT0303	OUT PWM-25-B
OUT0304	OUT PWM-25-A
OUT0305	OUT PWM-25-B
OUT0306	OUT PWM-40-Bridge-A
OUT0307	OUT PWM-40-Bridge-A
OUT0308	OUT PWM-40-A
OUT0400	OUT PWM-25-A
OUT0401	OUT PWM-25-B
OUT0402	OUT PWM-25-A
OUT0403	OUT PWM-25-B
OUT0404	OUT PWM-25-A
OUT0405	OUT PWM-25-B
OUT0406	OUT PWM-40-Bridge-A
OUT0407	OUT PWM-40-Bridge-A
OUT0408	OUT PWM-40-A

IEC identifier	Output type
OUT0500	OUT PWM-25-A
OUT0501	OUT PWM-25-B
OUT0502	OUT PWM-25-A
OUT0503	OUT PWM-25-B
OUT0504	OUT PWM-25-A
OUT0505	OUT PWM-25-B
OUT0506	OUT PWM-40-Bridge-A
OUT0507	OUT PWM-40-Bridge-A
OUT0508	OUT PWM-40-A
OUT3000	OUT Supply-A
OUT3001	OUT Voltage-A
OUT3002	OUT Voltage-A

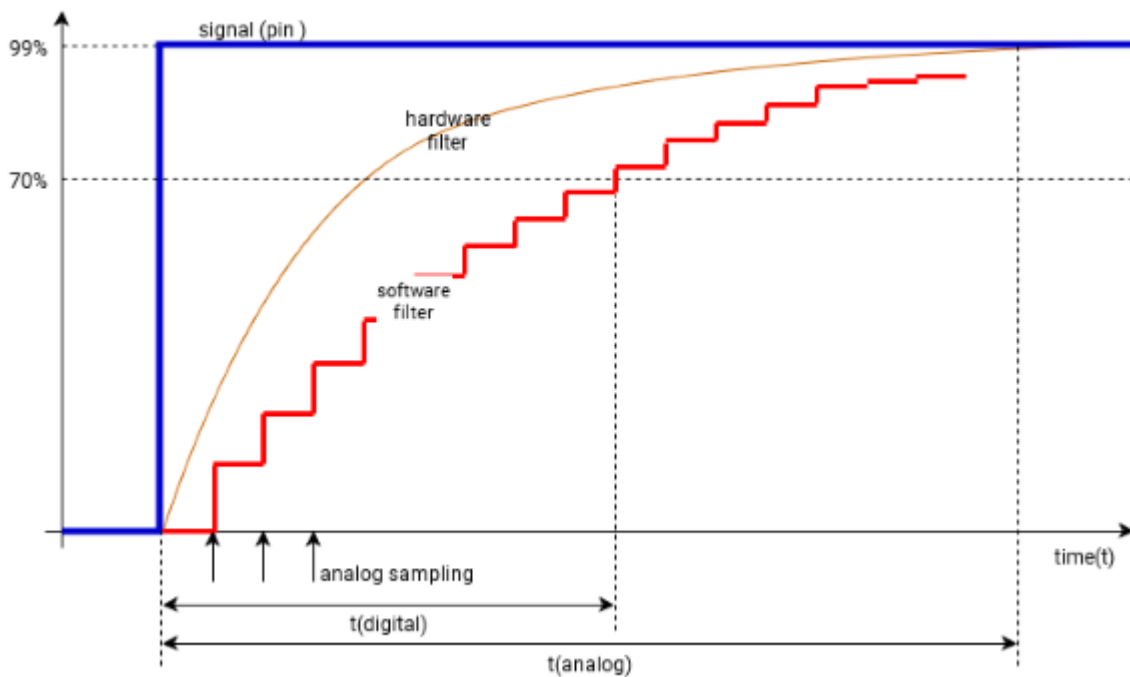
## 14.3 Filter

54552

Signals on analogue and digital inputs as well as the current measurements of the outputs are filtered by the controller. The filtering is done with the hardware filter (not configurable) and the first-order software filter (configurable in the IEC application).

By default, the switching thresholds for digital signals are 30% and 70% of VBB30. Therefore, the filter time  $t(\text{digital})$  for digital inputs is shorter than the filter time  $t(\text{analog})$  for analogue inputs.

The filters for analogue inputs and current measurements on the outputs are calculated with 99% of the signal change.





### 14.3.1 Filter times of the inputs

54553

Filters of the inputs				
Filter no.	t <sub>max</sub> (digital)	t <sub>nom</sub> (digital)	t <sub>max</sub> (analog)	t <sub>nom</sub> (analog)
0	0.7ms	0.6 ms	1.8ms	1.7ms
1	1.1ms	0.9ms	3.9ms	3.3ms
2	2.5ms	2.0ms	8.7ms	7.0ms
3	5.0ms	4.0ms	17.6ms	14.1ms
4	9.5ms	7.6ms	36.0ms	28.9ms
5	19.0ms	15.2ms	73.0ms	58.4ms
6	38.5ms	30.8ms	146.5ms	117.2ms
7	77.0ms	61.6ms	294.0ms	235.2ms
8	154.0ms	123.2ms	588.5ms	470.8ms
9	308.0ms	246.4ms	1178.0ms	942.4ms
10	616.5ms	493.2ms	2357.0ms	1885.6ms
11	1233.0ms	986.4ms	4715.0ms	3772.0ms
12	2465.5ms	1972.4ms	9430.5ms	7544.4ms

### 14.3.2 Filter times of the outputs

54554

Filter for current measurement of the output in digital operation		
Filter no.	t <sub>max</sub> (analog)	t <sub>nom</sub> (analog)
0	1.7ms	1.7ms
1	2.2ms	1.8ms
2	3.8ms	2.4ms
3	7.2ms	3.9ms
4	14.5ms	7.4ms
5	29.2ms	14.7ms
6	58.6ms	29.3ms
7	117.6ms	58.8ms
8	235.4ms	117.7ms
9	471.2ms	235.6ms
10	942.8ms	471.4ms
11	1886.0ms	943.0ms
12	3772.2ms	1886.1ms

## 14.4 Using the operators

58749



► Observe the CODESYS online help!

- → Online Help > CODESYS Development System > Reference, Programming > Operators



### WARNING

Unpredictable result when using operators (AND, OR, XOR, ...) with input values of different data types, for example BOOL and BYTE, etc.

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function.
- Only use operators with input values of the same data types.

The following table shows in which PLC and in which PRG CODESYS standard operators may be used:

Operator group	Operator	Standard PLC Standard PRG	Safety PLC Standard PRG	Safety PLC Safety PRG	Restriction
Arithmetic	ADD	X	X	X	
	SUB	X	X	X	
	MUL	X	X	X	
	DIV	X	X	X	
	MOD	X	X	X	
	MOVE	X	X	X	
	INDEXOF	X	X	X	
	SIZEOF	X	X	X	
Bit string	AND	X	X	X	
	OR	X	X	X	
	XOR	X	X	X	
	NOT	X	X	X	
	AND_THEN	X	X	X	
	OR_ELSE	X	X	X	
Bitshift	SHL	X	X	X	
	SHR	X	X	X	
	ROL	X	X	X	
	ROR	X	X	X	
Selection	SEL	X	X	X	
	MAX	X	X	X	
	MIN	X	X	X	
	LIMIT	X	X	X	
	MUX	X	X	X	

Operator group	Operator	Standard PLC Standard PRG	Safety PLC Standard PRG	Safety PLC Safety PRG	Restriction
Comparison	GT	X	X	X	
	LT	X	X	X	
	LE	X	X	X	
	GE	X	X	X	
	EQ	X	X	X	
	NE	X	X	X	
Address	ADR	X	X	X	
	Content Operator	X	X	X	
	BITADDR	X	X	X	
Call	CAL	X	---	---	
Type conversion	BOOL_TO	X	X	X	
	TO_BOOL	X	X	X	
	<INT_Type>_TO_ <INT_Type>	X	X	X	
	REAL_TO- / LREAL_TO	X	X	X	
	TIME_TO / TIME_OF_DAY_TO	X	X	X	
	DATE_TO / DT_TO	X	X	X	
	STRING_TO	X	X	X	
	TRUNC	X	X	X	
	TRUNC_INT	X	X	X	
	TO_<xxx>	X	X	X	
Numeric	ABS	X	X	X	For minimum of signed integer data types, the return value is undefined. (e.g. ABS(-128) for data type SINT.).
	SQRT	X	X	X	Definition range $\geq 0$ , For definition range $< 0$ Result = NaN
	LN DB	X	X	X	Definition range $> 0$ , For definition range $\leq 0$ Result = NaN
	LOG	X	X	X	Definition range $> 0$ , For definition range $\leq 0$ Result = NaN
	EXP	X	X	X	
	EXPT	X	X	X	
	SIN	X	X	X	For accuracy up to the 4th decimal place: Definition range (REAL) = -6.2678940e+01...1.3169797e+02
	ASIN	X	X	X	Definition range = -1.0...1.0
	COS	X	X	X	For accuracy up to the 4th decimal place:

Operator group	Operator	Standard PLC Standard PRG	Safety PLC Standard PRG	Safety PLC Safety PRG	Restriction
					Definition range (REAL) = -1.1476853e+02...1.1476853e+02
	TAN	X	X	X	For accuracy up to the 4th decimal place: REAL definition range = -4.5447968e+01...4.7023037e+01 Excepted are singularities around $\pi/2 + k \cdot \pi \pm 0.1$
	ACOS	X	X	X	Definition range = -1.0...1.0
	ATAN	X	X	X	
Namespace	Namespace for Global Variable Lists	X	X	X	
	Enumeration Namespace	X	X	X	
	Library Namespace	X	X	X	
Other	__DELETE	---	---	---	
	__ISVALIDREF	---	---	---	
	__NEW	---	---	---	
	__QUERYINTERFACE	---	---	---	
	__QUERYPOINTER	---	---	---	
	__TRY, __CATCH, __FINALLY, __ENDTRY	---	---	---	
	__VARINFO	---	---	---	
	INI	---	---	---	

Legend:

X = may be used

--- = may not be used

## 14.5 Behaviour in case of floating point operations

60628

REAL operations are calculated in the CPU with the Floating Point Unit (FPU). This leads to a fast execution of these arithmetic operations.

LREAL operations are calculated via software library. This leads to a longer processing time for the arithmetic operations.

The FPU supports IEEE-754 REAL values, except for denormalised REAL values.

► If denormalised values cannot be excluded in calculations: Use LREAL values.

## 14.5.1 Behaviour in case of specific arguments

60629

When calculating operations and functions with special REAL/LREAL arguments, the controller shows the following behaviour:

- REAL-/LREAL-division by +/- 0.0 leads to a `SERIOUS_ERROR` and to the state `RUNTIME_STOP`.
- REAL/LREAL operations with the argument NaN (not a number) have the result NaN. Exceptions:  $\text{expt}(1.0, \text{NaN}) = 1.0$ ;  $\text{expt}(\text{NaN}, 0.0) = 1.0$
- REAL/LREAL operations with the argument +/- Inf (Infinity) lead to undefined results or to a `SERIOUS_ERROR` and into the state `RUNTIME_STOP`.
- REAL operations with the arguments +/- denormal lead to undefined results.
- REAL operations `<=` and `>=` with at least one argument NaN lead to the result `TRUE` (IEEE Intel Issue) .



### WARNING

Unpredictable control behaviour when calling up mathematical operations and functions with special REAL/LREAL arguments

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function is possible.
- ▶ Only use REAL/LREAL arguments within the normal definition range for mathematical operations/functions.
- ▶ Avoid special REAL/LREAL arguments in calculations.
- ▶ Check REAL/LREAL arguments before calculations.

## 14.5.2 Behaviour in case of conversions

60631

The following behaviour occurs in the conversion functions REAL\_TO\_[TYPE] and LREAL\_TO\_[TYPE]:

Function	Argument	Result
REAL_TO_BYTE, REAL_TO_USINT, REAL_TO_WORD, REAL_TO_UINT, LREAL_TO_BYTE, LREAL_TO_USINT, LREAL_TO_WORD, LREAL_TO_UINT	< 0	SERIOUS_ERROR
REAL_TO_BYTE, REAL_TO_USINT, REAL_TO_WORD, REAL_TO_UINT, LREAL_TO_BYTE, LREAL_TO_USINT, LREAL_TO_WORD, LREAL_TO_UINT	> Upper value range limit [TYPE]	Undefined value or SERIOUS_ERROR
REAL_TO_SINT, REAL_TO_INT, LREAL_TO_SINT, LREAL_TO_INT	< Lower value range limit [TYPE] > Upper value range limit [TYPE]	Undefined value or SERIOUS_ERROR
REAL_TO_DWORD, REAL_TO_DINT, REAL_TO_UDINT, REAL_TO_LWORD, REAL_TO_ULINT, REAL_TO_LINT, LREAL_TO_DWORD, LREAL_TO_DINT, LREAL_TO_UDINT, LREAL_TO_LWORD, LREAL_TO_ULINT, LREAL_TO_LINT	< Lower value range limit [TYP] > Upper value range limit [TYP]	SERIOUS_ERROR



### WARNING

The result of the conversion functions REAL\_TO\_[TYPE] and LREAL\_TO\_[TYPE] does not fit into the value range of the target data type (e.g. BYTE, USINT, ...).

- > The corresponding PLC continues to run with incorrect values.
- > The corresponding PLC passes into the safe state.
- Ensure that no invalid conversions are performed.
- Before conversion, check for valid arguments within the value range of the target data type, see tables.

## 14.6 Behaviour in case of integer operations

60632

The controller shows the behaviour described in the table and leads to the described results if invalid integer arguments are used for calculation:

Operator	Arguments	Result
ABS	Minimum signed variable e.g. ABS(-128) (minimum of data type SINT)	Minimum signed variable -128
MOD	0	0

## 14.7 Mapping table [H2] user manual / ifm ecomatController CR7xxS

54556



### WARNING

If the safety requirements when programming the safety PLC with CODESYS SIL2 are not complied with.

- > Risk of personal injuries and/or damage to property.
- > Failure of the safety function.
- ▶ Adhere to the safety requirements according to the information in the table when creating safety-related applications with the controller.

The following table refers to the safety requirements in the [H2] user manual CODESYS Safety SIL 2 V6.0, chapter 7.4, by CODESYS GmbH.

The table shows which of the safety requirements are necessary when using a CR7xxS controller and which are not.

ID	Description	Reference	required	not required	Comment
§H2-1.1	Compliance with the safety requirements	S3-4.2	x	-	
§H2-2.1	Unsafe monitoring	[S2-4.7]	x	-	
§H2-2.2	Fail-safe logging	[S2-4.18]	x	-	
§H2-3.1	Approval of the IEC application	[S2-4.2]	x	-	
§H2-3.2	Review of the IEC application	[S2-4.17]	x	-	
§H2-3.3	Archiving of the IEC application	[S2-4.16]	x	-	
§H2-4.1	Double version	[S2-4.4]	-	x	
§H2-4.2	Double I/O image	[S2-4.4]	-	x	
§H2-4.3	Source Code Reviews	[S3-4.3]	x	-	
§H2-4.4	Warnings in the code	[S2-5.1]	x	-	
§H2-4.5	Using libraries	[S3-4.4], [S2-4.1]	x	-	
§H2-5.1	Permitted programming languages	[S3-4.5]	x	-	

§H2-5.2	Floating point operations	[S3-4.1]	x	-	With ifm, floating point operations can be used safe including the trigonometric functions
§H2-5.13	Non natural aligned structures	[S3-4.16]	x	-	
§H2-5.3	Multitasking and I/Os	[S2-4.5]	x	-	
§H2-5.4	Multiple mappings of I/Os	[S2-4.6]	x	-	
§H2-5.5	Permitted libraries	[S3-4.4], [S2-4.1]	x	-	
§H2-5.6	Validation of libraries	[S3-4.6]	x	-	
§H2-5.14	Using the operators AND_THEN / OR_ELSE	[S3-4.17]	x	-	
§H2-5.15	Using pragmas	[S3-4.15]	x	-	
§H2-5.7	Structured text as LVL	[S3-4.7]	x	-	
§H2-5.8	CFC: Module test required	[S3-4.10]	x	-	
§H2-5.9	CFC: Plating of inputs and outputs	[S3-4.11]	x	-	
§H2-5.10	CFC: Jumps are not allowed	[S3-4.12]	x	-	
§H2-5.11	UML/SC: Test of the state chart	[S3-4.13]	x	-	
§H2-5.12	UML/SC: Using the inputs Relnit / Abort	[S3-4.14]	x	-	
§H2-6.1	Calculation of fail-safe outputs	[S2-4.3]	x	-	
§H2-6.2	I/O devices without SAFE data types	[S3-4.8]	x	-	
§H2-6.3	Default values and normally closed operation	[S3-4.9]	x	-	
§H2-6.4	CANopen Safety Stack: Review of the configuration	[S2-4.8]	x	-	
§H2-6.5	CANopen Safety Stack: Cross communication	[S2-4.15]	x	-	
§H2-6.6	CANopen Safety Stack: Mapping of arrays, structures and partial mapping	[S2-4.9]	x	-	
§H2-6.7	CANopen Safety Stack: Evaluation of the diagnostics	[S2-4.10]	x	-	
§H2-6.8	CANopen Safety Stack: Monitoring and diagnostics in the device tree are unsafe	[S2-4.14]	x	-	
§H2-6.9	CANopen Safety Stack: Unwanted reconfiguration operating	[S2-4.11]	x	-	
§H2-6.10	CANopen Safety Stack: Demo mode is not possible	[S2-4.12]	-	x	The licences are included in all controllers
§H2-6.11	CANopen Safety Stack: GFCs are not fail-safe	[S2-4.13]	x	-	

Legend:

x = applies

- = does not apply



## 14.8 Directory structure and file overview

39515

The following directories and files are stored in the device:

Data name / path	Description
<b>apps</b>	Folder
▪ standard.app	Application non-safe
▪ safe.app	Application safe
<b>os</b>	Folder
▪ ifmOS.ifm	ifmOS
<b>boot</b>	Folder
▪ boot.ifm	Bootloader
<b>sis</b>	Folder
▪ sissys.ifm	SIS-SYS
▪ sisdev1.ifm	SIS-DEV1
▪ sisdev2.ifm	SIS-DEV2
<b>cfg</b>	Folder
▪ comconf.cfg	Communication configuration
▪ memconf.ifm	Memory configuration
▪ iomapping.cfg	IO resource allocation
<b>calib</b>	Folder
▪ calib-sup.ifm	Calibration file
▪ calib-in.ifm	Calibration file
▪ calib-group.ifm	Calibration file
▪ calib-out.ifm	Calibration file
<b>retain</b>	Folder
▪ standard.ret	Application retain non-safe
▪ standard.mb	Application memory bytes non-safe
▪ safe.ret	Application retain safe
▪ safe.mb	Application memory bytes non-safe
<b>log</b>	Folder

Data name / path	Description
▪ logging.ifm	Log file
▪ statistics.ifm	Statistics file
<b>data</b>	Folder
▪ *.*	Memory space for user-defined data
<b>auth</b>	Folder
▪ auth.txt	Authentication file
<b>compat</b>	Folder
▪ compat.ifm	Compatibility File
<b>cmd</b>	Folder
▪ cmd.ifm	Command File
<b>info</b>	Folder
▪ appinfo.txt	Application information
▪ devinfo.txt	Device information
▪ swinfo.txt	Software information
▪ comlog.txt	Log file communication
▪ filelist.txt	List of files in folder /data

## 14.9 Overview of user rights

39386

	Both applications	Safety application		Standard application		IEC application SysFile
<b>User</b>	admin	s-admin	s-service	n-admin	n-service	iec
<b>Groups</b>	Administrator	s-admin	s-service	n-admin	n-service	iec
<b>Hidden</b>	no	no	no	no	no	yes
<b>Data system objects</b>						
/	ro	ro	ro	ro	ro	ro
/os	rw	ro	ro	ro	ro	ro
/boot	rw	ro	ro	ro	ro	ro
/sis/sissys.ifm	rw	ro	ro	ro	ro	ro
/cfg	rw	ro	ro	ro	ro	ro
/security	rw	na	na	na	na	na
/apps/safe.app	rw	rw	ro	na	na	ro
/apps/standard.app	rw	na	na	rw	ro	ro
/retain/safe.ret	rw	rw	ro	na	na	ro
/retain/safe.mb	rw	rw	ro	na	na	ro
/retain/standard.ret	rw	na	na	rw	ro	ro
/retain/standard.mb	rw	na	na	rw	ro	ro
/log/logging.ifm	ro	ro	ro	ro	ro	ro
/data	rw	rw	ro	rw	ro	rw
/compat	rw	rw	ro	rw	ro	rw
/cmd	rw	na	na	na	na	rw
/info	ro*	ro*	ro*	ro*	ro*	ro*
<b>Processing time objects</b>						
UserManagement	AECV	V	V	V	V	na
<b>Safety PLC</b>						
Each sub-node	AECV	AECV	EV	na	na	na
<b>Standard PLC</b>						
Each sub-node	AECV	na	na	AECV	EV	na

ifm rights		Authorised access
ro	read only	read only
ro*	read only	read only; also possible without user login
rw	read write	read and write
na	not available	not allowed

CODESYS rights				Permissions			
(from CODESYS online help)				A	E	C	V
Objects			Action	Add/ remove children	Execute	Change	View
Device			Login	-	-	-	x
	Logger		Read entries	-	-	-	x
	PlcLogic						
		Application	Login	-	-	-	x
			Create	-	-	x	-
			Create child object	x	-	x	-
			Delete	-	-	x	-
			Load / online change	-	-	x	-
			Create boot project	-	-	x	-
			Read variable	-	-	-	x
			Write variable	-	-	x	x
			Force variable	-	-	x	x
			Set and delete breakpoint	-	x	x	-
			Set next statement	-	x	x	-
			Read call stack	-	-	-	x
			Single cycle	-	x	-	-
			Switch on flow control	-	x	x	-
			Read flow control	-	-	-	x
			Start / Stop	-	x	-	-
			Reset	-	-	x	-
	PlcShell		Execute PLC shell command	-	x	-	-
	Settings		Read settings	-	-	-	x
			Write settings	-	-	x	-
	UserManagement		Read configuration	-	-	-	x
			Write configuration	-	-	x	-

## 14.10 Task configuration example

54558

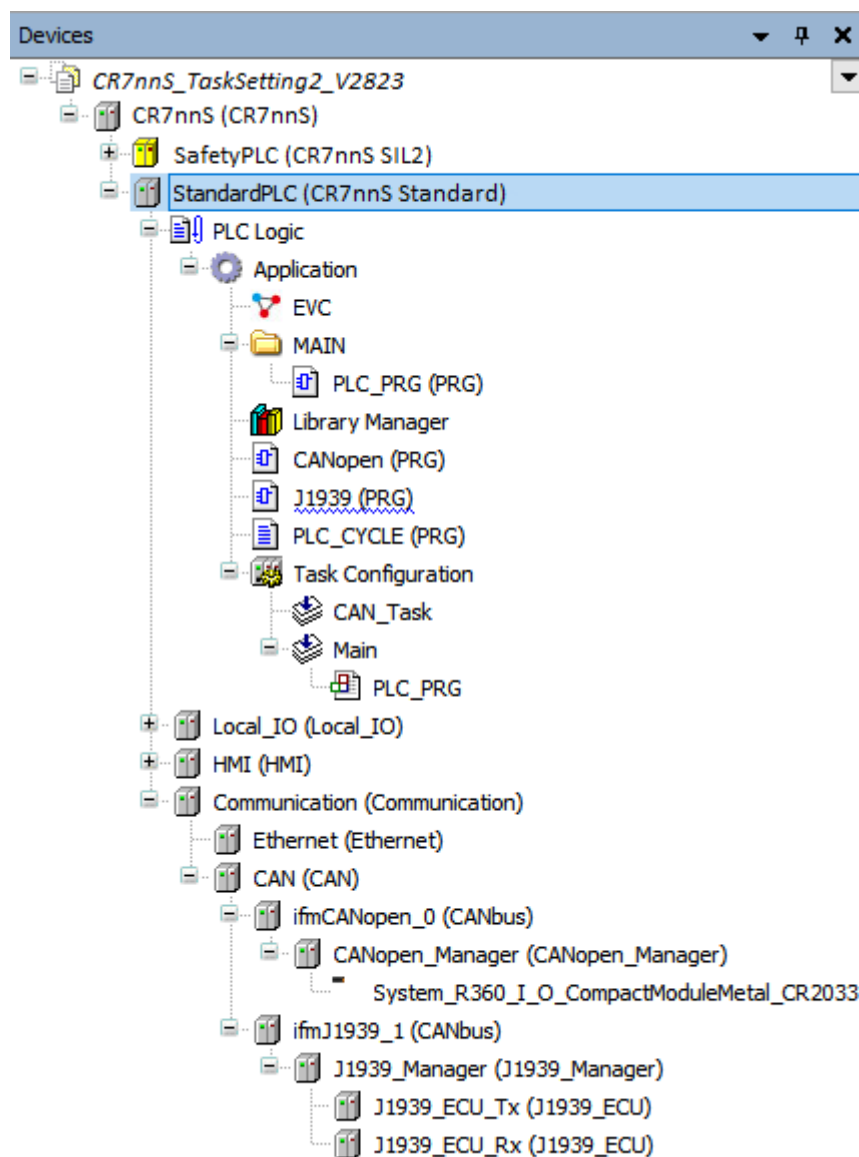
The following example describes an option to configure task and bus settings of a CR7xxSCODESYS project

The example does not claim to be exhaustive and to apply to all cases, but merely illustrate the technical aspects.

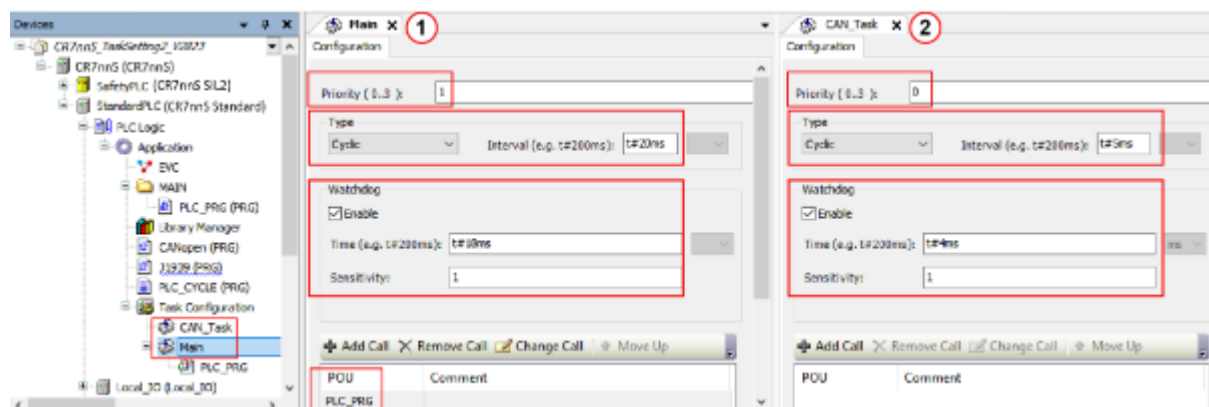
Whether the presented exemplary configuration can be transferred to the respective application must be evaluated and verified by the system builder.

### 14.10.1 Current situation

- CODESYS project with CR7xxS
- The device tree of the StandardPLC is opened
- The CAN interface 0 is configured as ifmCANbus / CANopen
- ifm CR2033 CompactModule is configured as a node on CAN interface 0
- CAN interface 1 is configured as ifmCANbus / J1939
- J1939\_ECU\_Tx is configured as local device on CAN interface 1
- J1939\_ECU\_Rx is configured as node on CAN interface 1
- In the application of the StandardPLC, the following PRGs are available
  - CANopen (CANopen processing)
  - J1939 (J1939 processing)
  - PLC\_PRG (main program)



## 14.10.2 Setting the task configuration

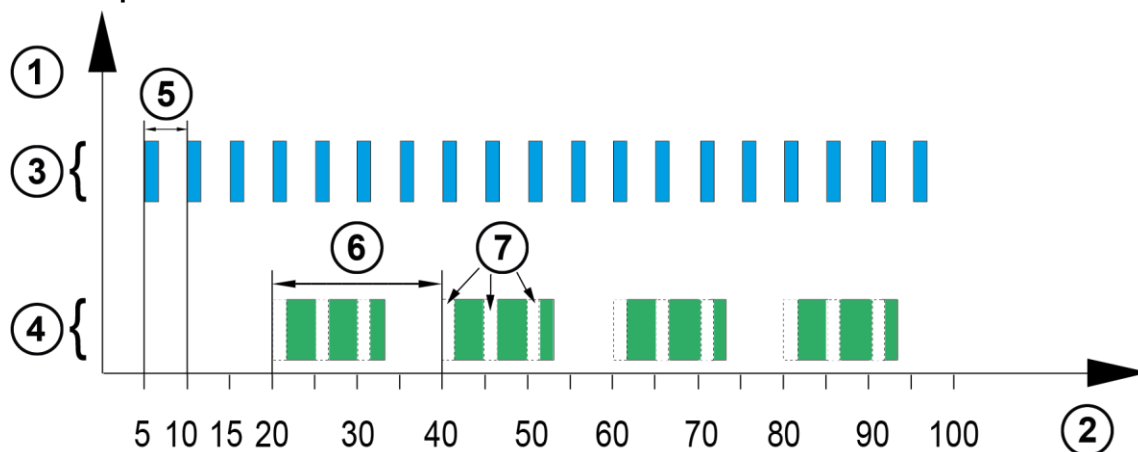


► Create the following tasks in the task configuration of the StandardPLC and configure them according to the table:

- 1: Main for the main program (PLC\_PRG)
- 2: CAN\_Task for data transmission via CAN bus

Task	Priority	Type	Interval	Watchdog enabled	Watchdog time	Watchdog sensitivity	POU call
Main	1	Cyclic	t#20ms	activated	t#18ms	1	PLC_PRG
CAN_Task	0 (highest)	Cyclic	t#5ms	activated	t#4ms	1	--

**Time response of the tasks**



- 1: Priority
- 2: Time
- 3: CAN\_Task with priority 0 (highest)
- 4: Main task with priority 1
- 5: Interval time CAN\_Task = 5 ms
- 6: Interval time main task = 20 ms
- 7: Pre-emptive behaviour: Interruption of the execution of the main task by CAN\_Task of higher priority

**Information of task settings:**

- The task behaviour of the controller is generally pre-emptive. This means that the task with the higher priority interrupts the task with the lower priority if the task with the lower priority is not yet processed in case of a renewed interval start of the task with the higher priority. → **Time-related behaviour IEC application** (→ p. [33](#))
- ▶ Recommendation:
  - A maximum of 4 tasks is possible. However, ideally only use 2 tasks since the complexity of the time response and the necessary task settings increases with the number of tasks due to the pre-emptive behaviour.
  - Division into 2 tasks: Task 1 for CAN communication, task 2 to process the main program. Possible settings, see table.
- ▶ For error-free CAN communication:
  - Using 2 tasks: Process the CAN communication in an own task.
  - Interval time of the CAN task: According to the J1939 specification, the Transmission Repetition Rate for TSC1 must be 10 ms. The interval for the CAN task is set to 5 ms. Thanks to this, despite of the fluctuation, (Jitter = approx. 1 ms) it is assured that the Transmission Repetition Rate of 10ms will not be exceeded and no messages will be lost.
  - CAN task priority: Assign the highest priority to the CAN task to assure cyclic execution.



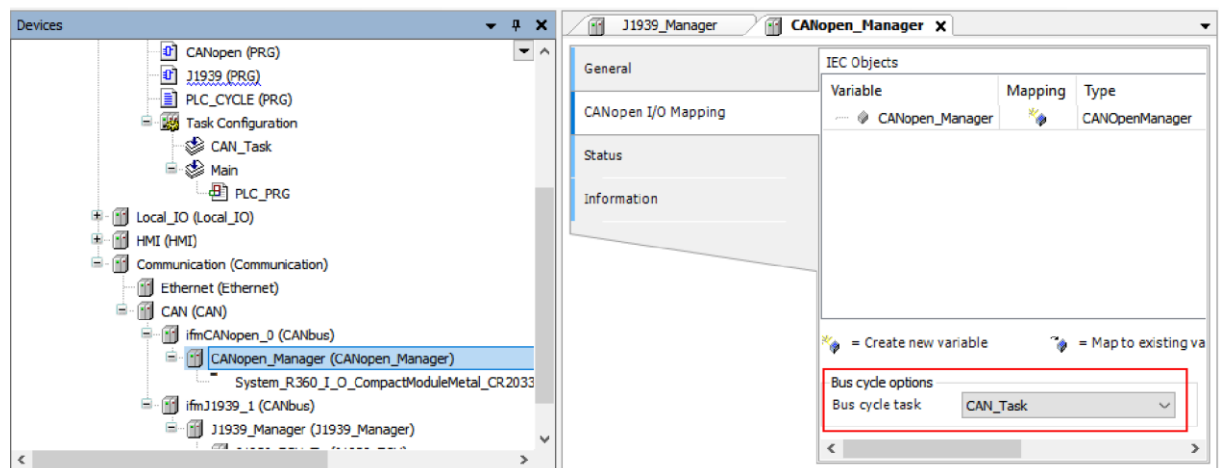
### 14.10.3 Assign the CAN manager to the bus task

54561

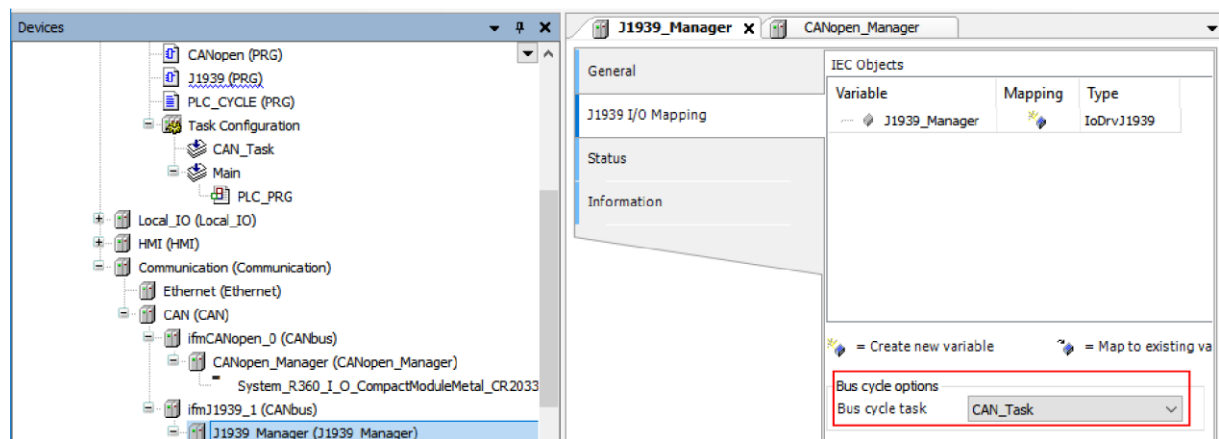
In the properties of the CANopen / J1939 manager, the CANopen / J1939 I/O image is assigned to the CAN\_Task.

Only if the assignment is correct, the CAN communication will run in the CAN task interval.

#### 1. Assignment of the CANopen I/O image to the CAN\_Task



#### 2. Assignment of the J1939 I/O image to the CAN\_Task



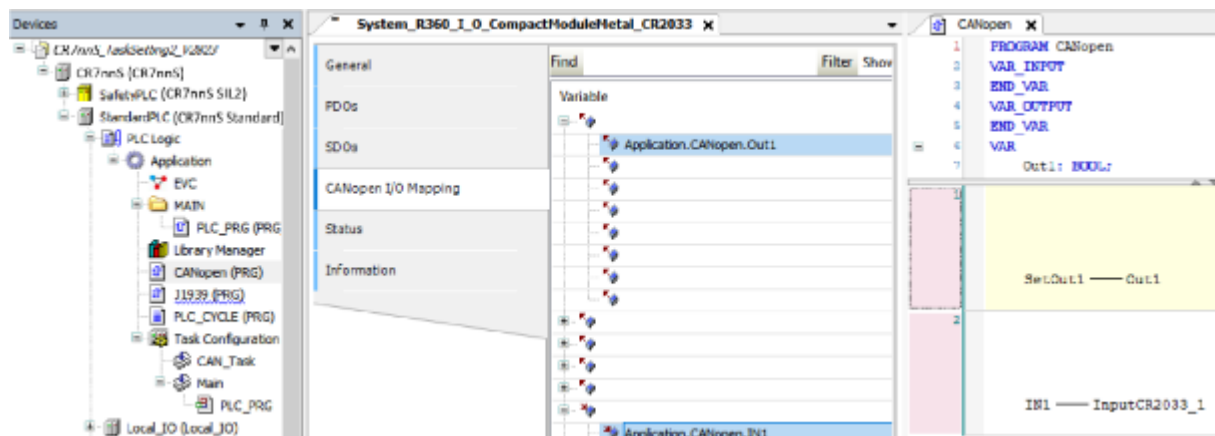
## 14.10.4 Mapping of the CAN variables

The CAN variables are mapped on PRG variables.

The values of the CAN variables are accessed from the PRG and the PRG variables.

### Mapping in the I/O image CANopen

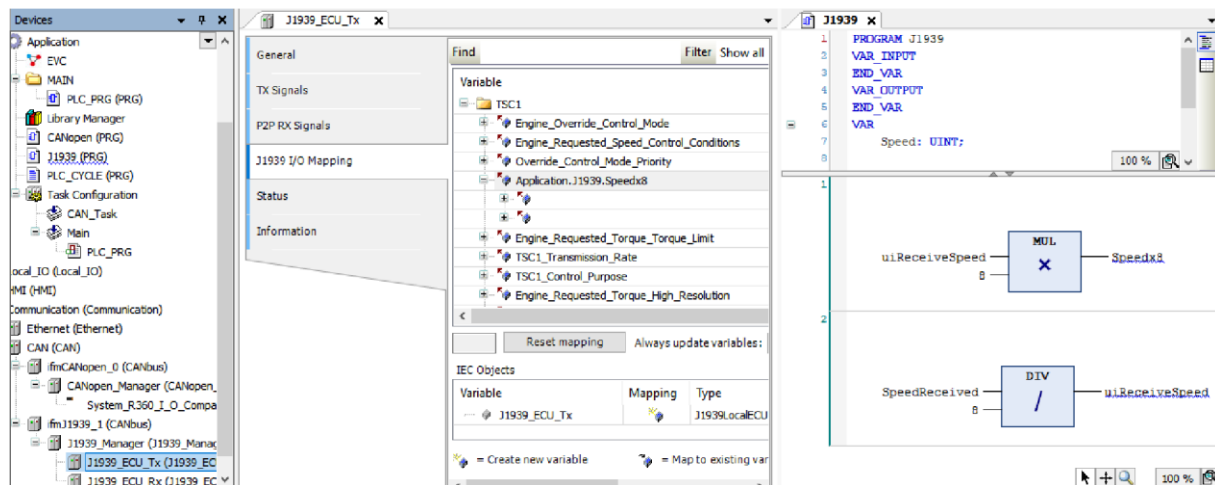
CAN variable channel	PRG variable	Application	PRG name	Variable name in the PRG
Outputs1.Bit0	Application.CANopen.Out1	Application	CANopen	Out1
Inputs1.Bit0	Application.CANopen.IN1	Application	CANopen	IN1



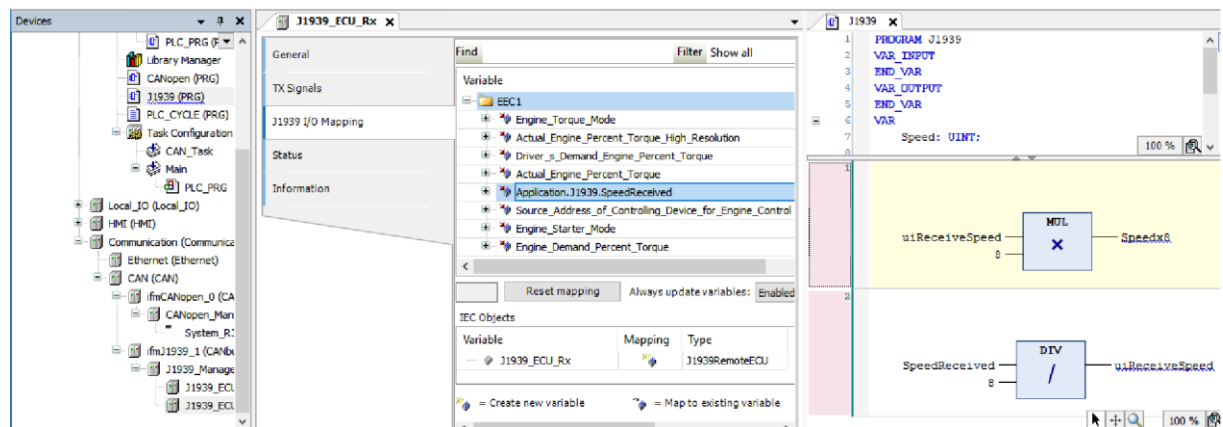
### Mapping in the I/O image J1939

Device	CAN variable channel	PRG variable	Application	PRG name	Variable name in the PRG
J1939_ECU_Tx	TSC1.Engine_Requested_Speed_Speed_Limit	Application.J1939.Speed8x	Application	J1939	Speed8x
J1939_ECU_Rx	EEC1.Engine_Speed	Application.J1939.SpeedReceived	Application	J1939	SpeedReceived

### J1939\_ECU\_Tx



## J1939\_ECU\_Rx



## 14.10.5 Call CAN POU's

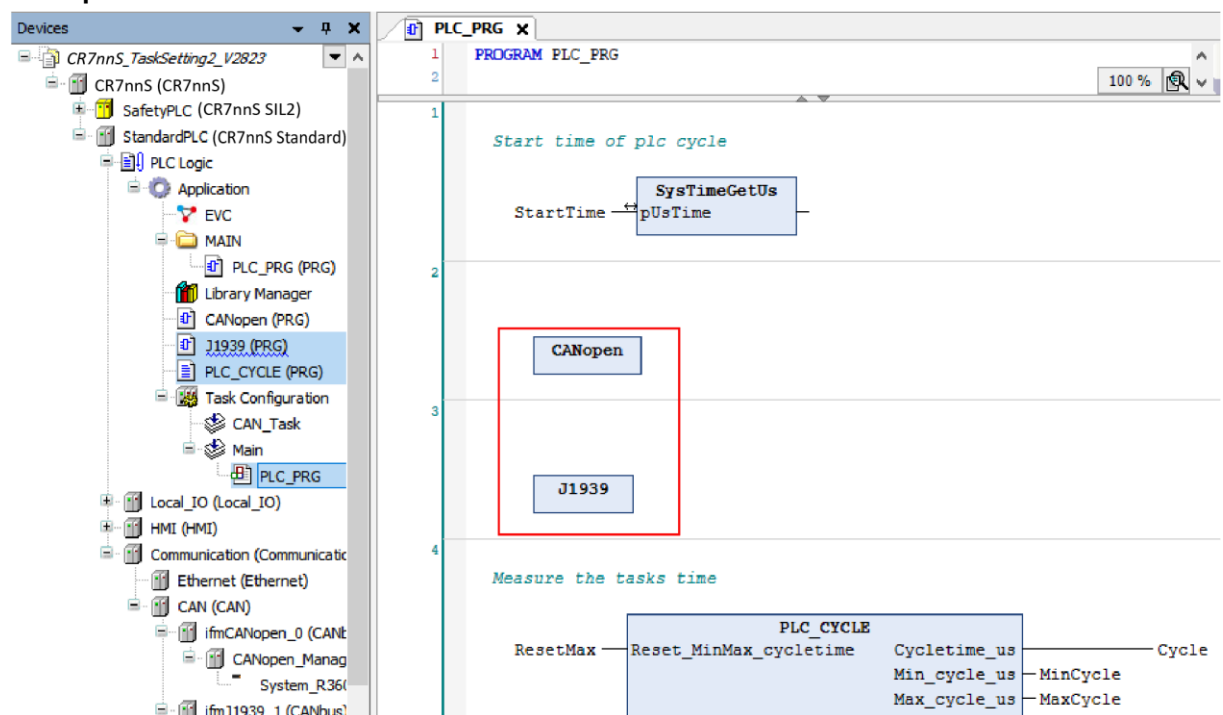
54563

The CAN variables are further processed in the PRGs CANopen and J1939. Content of the PRGs (programming example) → [Mapping of the CAN variables](#) (→ p. 494)

The main program PLC\_PRG calls the PRGs.

All PRGs are processed in the main task.

### Call up of the PRGs



## 14.10.6 Task monitoring

54564

### Online monitoring of the bus cycle time

The screenshot shows the CODESYS Task Configuration window for a project named 'CR7nnS\_TaskSetting2\_V2023'. The window is divided into two main panes: 'Properties' and 'Monitor'. The 'Monitor' pane is active, displaying a table of task execution data.

Task	Status	IEC-Cycle Count	Cycle Count	Last Cycle Time (µs)	Average Cycle Time (µs)	Max. Cycle Time (µs)	Min. Cycle Time (µs)
CAN_Task	Valid	35066	35066	650	671	1805	627
Main	Valid	8913	8913	249	454	2135	199

The left pane shows the project tree with the following structure:

- CR7nnS (CR7nnS)
- SafetyPLC (CR7nnS SIL2)
- PLC Logic
  - Application
    - EVL
    - Library Manager
    - PLC\_PRG (PLC)
    - Task Configuration
      - Task
        - PLC\_PRG
  - Local\_IO (Local\_IO)
  - HMI (HMI)
  - Communication (Communication)
  - StandardPLC [connected]
  - PLC Logic
  - Application [run]

## 14.11 ifm behaviour models for function blocks

### Content

General .....	497
Behaviour model ENABLE .....	497
Behaviour model EXECUTE .....	498

39475

This chapter describes the ifm behaviour models for function blocks.

### 14.11.1 General

39507

ifm function blocks feature the following outputs to return status and error information:

Output	Description	
xError	TRUE	An error has occurred.
	FALSE	No error has occurred.
eDiagInfo	Diagnostic/error information → <b>Messages / diagnostic codes of the function blocks</b> (→ p. <a href="#">463</a> )	

All inputs and outputs in the function block that belong to the ifm behaviour model are featured at the top.

### 14.11.2 Behaviour model ENABLE

39578

Function blocks that use the behaviour model ENABLE are cyclically processed as long as the status at the input is xEnable = TRUE.

If xEnable = FALSE, the function block will not be executed. All function block outputs are reset to their preset default values and will not be updated. In this case the following applies: xError = FALSE and eDiagInfo = STAT\_INACTIVE.

Function blocks that have no xEnable input are processed cyclically when the application is started. The processing is only terminated when the application is stopped. The behaviour corresponds with the behaviour of a function block with a permanent TRUE at the xEnable input.

### Response to errors

39672

In case of an error, xError is set to TRUE and eDiagInfo indicates the diagnostic code as long as xEnable is = TRUE.

Irrespective of the data type, all other outputs of the function block will be reset to the following values:

Data type	Value
numerical	0 / 0.0
String	Empty string
BOOL/Bit	FALSE

### 14.11.3 Behaviour model EXECUTE

39579

Function blocks that have the EXECUTE behaviour model are processed once after a rising edge at the xExecute input.

If the function block has executed its function successfully, the output is set xDone = TRUE.



For some function blocks, the signal to xExecute has to remain set to TRUE until xDone=TRUE. If the signal to xExecute becomes FALSE beforehand, the edit process of the function block is aborted without result.

The performance can be seen from the description of the input parameter xExecute of the corresponding function block.

### Response to errors

39597

In case of an error, xError is set to TRUE and eDiagInfo indicates the error status as long as xExecute is = TRUE.

The output xDone is set to FALSE since the execution could not be finished successfully.

Irrespective of the data type, all other outputs of the function block will be reset to the following values:

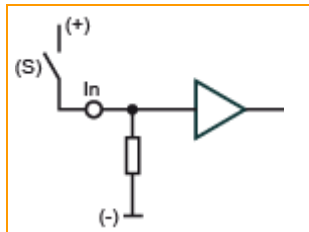
Data type	Value
numerical	0 / 0.0
String	Empty string
BOOL/Bit	FALSE

## 15 Terms and abbreviations

### C

#### CSI

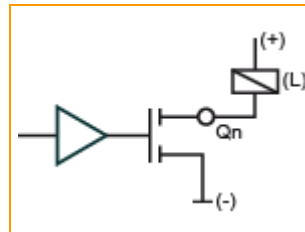
CSI = Current Sinking



**Current Sinking Input**

In = connection of binary input n  
(S) = sensor

Binary input block diagram, plus-switching  
for positive sensor signal  
Input = open  $\Rightarrow$  Signal = Low (GND)  
 $\rightarrow$  Low-Side Input ( $B_L$ )



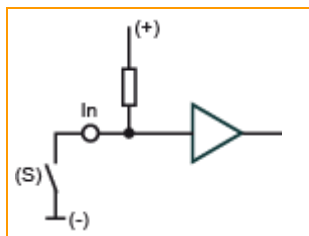
**Current Sinking Output**

Qn = connection of output n  
(L) = load

Output block diagram, minus-switching ( $B_L$ )  
for negative output signal  
Low-Side Output ( $B_L$ )

#### CSO

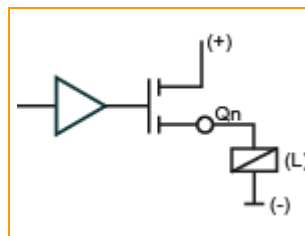
CSO = Current Sourcing



**Current Sourcing Input**

In = connection of binary input n  
(S) = Sensor

Binary input block diagram, minus-switching  
for negative sensor signal:  
Input = open  $\Rightarrow$  Signal = High (Supply)  
 $\rightarrow$  High-Side Input ( $B_H$ )



**Current Sourcing Output**

Qn = connection of output n  
(L) = load

Output block diagram, plus-switching  
for positive output signal  
High-Side Output ( $B_H$ )

### D

#### Dither

Dither is a component of the  $\rightarrow$  PWM signals to control hydraulic valves. It has shown for electromagnetic drives of hydraulic valves that it is much easier for controlling the valves if the control signal (PWM pulse) is superimposed by a certain frequency of the PWM frequency. This dither frequency must be an integer part of the PWM frequency.

### O

#### Operating system

Basic program in the device that establishes the connection between the hardware of the device and the application program.

$\rightarrow$  Chapter **Software module for the device** ( $\rightarrow$  p. [62](#))

## P

### Process image

Process image is the status of the inputs and outputs the PLC operates with within one →cycle.

- At the beginning of the cycle the PLC reads the conditions of all inputs into the process image. During the cycle the PLC cannot detect changes to the inputs.
- During the cycle the outputs are only changed virtually (in the process image).
- At the end of the cycle the PLC writes the virtual output states to the real outputs.

### PWM

PWM = pulse width modulation

The PWM output signal is a pulsed signal between GND and supply voltage.

Within a defined period (PWM frequency) the mark-to-space ratio is varied. Depending on the mark-to-space ratio, the connected load determines the corresponding RMS current.

## R

### ratiometric

Measurements can also be performed ratiometrically. If the output signal of a sensor is proportional to its supply voltage then via ratiometric measurement (= measurement proportional to the supply) the influence of the supply's fluctuation can be reduced, in ideal case it can be eliminated.

→ analogue input



# 16 Index

## 1

1-channel safety concept for inputs ..... 148

## 2

2-channel safety concept for inputs ..... 157

## A

About this manual ..... 7  
 Access inputs ..... 125  
 Access outputs ..... 126  
 Access protection ..... 17  
 Access to inputs ..... 123  
 Access to outputs ..... 124  
 Access to safety input data and safety output data ..... 144  
 Accessing the system inputs ..... 126  
 Accessing the system outputs ..... 127  
 Accessing user LEDs ..... 127  
 Activate the access protection for a project ..... 80  
 Add function libraries to the application ..... 93  
 Add function libraries to the safety application ..... 95  
 Address assignment in Ethernet networks ..... 102  
 Analogue input, 2-channel  
   Example, connector A ..... 159  
   Example, connector B ..... 160  
 Appendix ..... 475  
 Application ..... 66  
 Application information (appinfo.txt) ..... 200  
 Assign inputs/outputs - safety / standard PLC ..... 86  
 Assign memory allocation -- safety / standard PLC ..... 86  
 Assign the CAN manager to the bus task ..... 500  
 aSysInfoList (GVL) ..... 340  
 Available memory ..... 43

## B

Behaviour in case of conversions ..... 490  
 Behaviour in case of floating point operations ..... 489  
 Behaviour in case of integer operations ..... 491  
 Behaviour in case of specific arguments ..... 489  
 Behaviour in case of voltage dip ..... 46  
 Behaviour model ENABLE ..... 505  
 Behaviour model EXECUTE ..... 506  
 Block diagram of the supply and of the output deactivation ..... 48  
 Bootloader ..... 65

## C

Call CAN POU ..... 503  
 Call sequence for diagnostic function blocks for inputs and outputs .. 122  
 CAN  
   Interfaces and protocols ..... 63  
 CAN libraries ..... 228  
 CAN\_BAUDRATE (ENUM) ..... 222  
 CAN\_BUS\_STATE (STRUCT) ..... 254  
 CAN\_CHANNEL (ENUM) ..... 222  
 CAN\_Enable ..... 230  
 CAN\_Info (GVL) ..... 254  
 CAN\_Recover ..... 232  
 CAN\_RemoteRequest ..... 234

CAN\_RemoteResponse ..... 236  
 CAN\_Rx ..... 238  
 CAN\_RxMask ..... 241  
 CAN\_RxRange ..... 244  
 CAN\_RxRangeExt ..... 247  
 CAN\_Status ..... 249  
 CAN\_Tx ..... 252  
 CANconstants (GVL) ..... 222  
 CANopen  
   Network Management (NMT) ..... 130  
   Send and receive SDO ..... 130  
 CANopen safety error behaviour ..... 182  
 Carry out installation ..... 71  
 Change history ..... 10  
 Characteristic safety values for CANopen Safety ..... 183  
 Check the communication path (blinking test) ..... 80  
 Check the hardware version of the device ..... 188  
 Check the operating system version of the device ..... 188  
 CODESYS ..... 10  
 CODESYS Debugging ..... 203  
 CODESYS Development System ..... 71  
 Complete package for ecomatController CR7xxS ..... 71  
 Complexity level / user level ..... 138  
 Components of the software package ..... 71  
 ConfigDiagLevel ..... 308  
 ConfigDiagProt ..... 311  
 ConfigSwThreshold ..... 266  
 Configuration of the safety task processing ..... 95  
 Configuration time ..... 40  
 Configure Ethernet interface ..... 101  
 Configure IEC watchdog ..... 99  
 Configure inputs and outputs ..... 111  
 Configure interfaces ..... 101  
 Configure serial interface ..... 101  
 Configure task processing ..... 92  
 Configure the programming interface ..... 79  
 Configuring CAN interfaces ..... 103  
 Configuring the safety PLC ..... 94  
 Configuring the standard PLC ..... 91  
 Connect the device to the network ..... 187  
 Control device ..... 124  
 Controlling LEDs in the applications ..... 212  
 COP\_GetNodeState ..... 256  
 COP\_SDOread ..... 258  
 COP\_SDOwrite ..... 260  
 COP\_SendNMT ..... 262  
 COUNT\_DIRECTION (ENUM) ..... 278  
 Create a network diagram ..... 198  
 Create a user in the CODESYS project ..... 84  
 Create CODESYS project ..... 76  
 Create new project with CR7xxS ..... 76  
 Creating a safety PLC application ..... 137  
 Creating a standard PLC application ..... 116  
 Cross references inputs ..... 50  
 Cross references outputs ..... 55  
 CSI ..... 507  
 CSO ..... 507  
 Current situation ..... 497  
 CurrentControl ..... 329  
 Cyclic diagnostics ..... 30

**D**

Data exchange between standard PLC and safety PLC .....	99
Data transmission for series production .....	194
Data transmission with TFTP .....	196
Data transmission with the ifm Maintenance Tool .....	195
Data types .....	116
Default behaviour of the inputs and outputs in case of an error .....	120, 144
Delete the application program on the device .....	192
description .....	416
Description 226, 230, 232, 234, 236, 239, 242, 244, 247, 250, 252, 256, 258, 260, 262, 266, 270, 272, 275, 285, 288, 291, 293, 295, 297, 308, 311, 317, 323, 329, 332, 339, 344, 349, 353, 357, 361, 365, 370, 376, 379, 382, 385, 388, 392, 394, 396, 399, 402, 405, 411, 413, 420, 424, 427, 430, 432, 435, 438, 441, 444, 447, 450, 453, 457, 460, 463, 466 .....	
DEST (ENUM) .....	314
Detailed error evaluation .....	162, 166
Device library .....	220
Device supply (technology) .....	45
Diagnostic Test Intervall (DTI) .....	35
Diagnostics .....	30
Digital input block diagram plus/minus-switching .....	51
Digital input, 2-channel .....	
Example, connector A .....	158
Example, connector B .....	159
Digital output block diagram plus/minus-switching .....	56
Directory structure and file overview .....	494
Display system information .....	202
Dither .....	507
Document the checksums .....	198
Document the serial number .....	198
Documentation of the overall application .....	198

**E**

eDIAG_PROT_MODE (ENUM) .....	315
ENCODER_RESOLUTION (ENUM) .....	279
Enter information about applications .....	85
Error classes .....	472
Error codes .....	356, 360, 364, 368, 373, 397
Error in the IEC application .....	114, 136
Error messages .....	472
Ethernet interface .....	62
Example .....	150, 153, 156, 162, 165, 168, 175, 176, 178, 180, 184
Executing reset variants .....	213

**F**

fail-safe 2-channel evaluation of the inputs .....	158
FastCount .....	270
Feedback in case of externally supplied outputs .....	60
File system - read/write performance .....	90
Files for series production .....	196
Filter .....	485
Filter times of the inputs .....	485
Filter times of the outputs .....	486
FILTER_INPUT (ENUM) .....	299
FILTER_OUTPUT (ENUM) .....	300
FILTER_OUTPUT_GROUP (ENUM) .....	320
FREQ_SENSE_PERIODS (ENUM) .....	280
Function description of non-volatile data .....	119, 142
Functions and features .....	20

**G**

General .....	505
General information .....	216
GetInfo .....	339
Getting started .....	75
Group designations .....	42

**H**

Hardware .....	70
Hardware description .....	41
Hardware information (devinfo.txt) .....	200
Hardware structure .....	42
HBridge .....	323
Help function libraries .....	337

**I**

ifm behaviour models for function blocks .....	505
ifm function libraries .....	216
Important standards .....	15
IncEncoder .....	272
Influence of actuators on the output diagnostics .....	60
Input .....	285
Input and output libraries .....	264
input parameter .....	295
Input parameter 226, 230, 232, 234, 236, 239, 242, 250, 260, 262, 267, 272, 293, 297, 309, 312, 317, 323, 329, 332, 339, 344, 349, 376, 379, 382, 388, 392, 394, 399, 402, 405, 411, 413, 416, 420, 424, 427, 430, 432, 435, 438, 441, 444, 447, 450, 453, 457, 460, 463, 466 .....	
Input parameters 245, 247, 252, 256, 258, 270, 275, 285, 288, 291, 354, 358, 362, 366, 371, 385, 396 .....	
Input type IN DIGITAL-A/B .....	54
Input type IN FREQUENCY-B .....	52
Input type IN MULTIFUNCTION-A .....	51
Input type IN RESISTOR-A .....	53
Inputs (technology) .....	50
Install CODESYS Development System .....	71
Install package (PC/laptop) .....	72
Installation .....	69
Interface configuration file comconf.cfg .....	110
Interfaces .....	62
IT safety .....	17

**L**

LED_COLOUR (ENUM) .....	224
LED_FLASH_FREQ (ENUM) .....	225
Legal and copyright information .....	7
Libraries .....	66
Libraries for safety programming .....	67
Library ifmCANopenManager.library .....	255
Library ifmConfigSwThreshold.library .....	265
Library ifmDeviceCR07nn.library .....	221
Library ifmFastInput.library .....	269
Library ifmIOcommon.library .....	284
Library ifmIOconfigDiagProt.library .....	307
Library ifmIOSafety.library .....	343
Library ifmOutGroup .....	316
Library ifmOutHBridge .....	322
Library ifmOutPWM .....	328
Library ifmPLCopenAddonSafe.library .....	391
Library ifmPLCopenSafe.library .....	410

Library ifmRawCAN.library .....	229
Library ifmSysInfo.library .....	338
Licensing .....	64
List of inputs .....	475
List of outputs .....	480, 483
List of outputs CR720S .....	482
List of the inputs CR710S .....	475
List of the inputs CR711S .....	476
List of the inputs CR720S .....	477
List of the inputs CR721S .....	478
List of the outputs CR710S .....	480
List of the outputs CR711S .....	481
Load the safety application to the device .....	192
Load the standard application to the device .....	191

## M

Manage files .....	89
Managing device users .....	83
Mapping of the CAN variables .....	501
Mapping table [H2] user manual / ifm ecomatController CR7xxS .....	491
Memory allocation .....	43
Memory allocation variants .....	44
Memory protection .....	97
Messages / diagnostic codes of the function blocks .....	472
Modbus .....	133
MODE_BRAKE (ENUM) .....	326
MODE_CURRENT_CONTROL (ENUM) .....	335
MODE_FAST_COUNT (ENUM) .....	281
MODE_HBRIDGE (ENUM) .....	327
MODE_INC_ENCODER (ENUM) .....	282
MODE_INPUT (ENUM) .....	301, 303
MODE_OUTPUT (ENUM) .....	302
MODE_OUTPUT_GROUP (ENUM) .....	321
MODE_PERIOD (ENUM) .....	283
MODE_PWM (ENUM) .....	336
MODE_SYSTEM_SUPPLY (ENUM) .....	304
MODE_TEMPERATURE (ENUM) .....	305

## N

NMT_SERVICE (ENUM) .....	263
NMT_STATES (ENUM) .....	263
Non-volatile data .....	118, 141
Non-volatile safety-related data (dynamic) .....	143
Non-volatile safety-related data (static) .....	143
Note on wiring .....	42
Note! .....	16

## O

Objects in a safety PLC application .....	135
Objects of a standard PLC application .....	115
Operating states .....	207
Operating system .....	66, 507
Operation .....	206
Operation as 2-channel fail-safe analogue input .....	163
Operation as 2-channel fail-safe digital input .....	160
Operation as 2-channel fail-safe frequency input .....	167
Operation as fail-safe analogue input .....	151
Operation as fail-safe digital input .....	148
Operation as fail-safe digital input with blanking pulses .....	154
Operation as fail-safe digital output .....	170

Operation as PWM output and current-controlled output .....	172
Options to access the input data and output data .....	120
Output .....	288
Output 2-channel, safe, with output group .....	175
Output groups .....	43
output parameter .....	345
Output parameter .....	226, 232, 236, 239, 243, 253, 317, 324, 330, 333, 339, 350, 380, 383, 386, 389, 394, 400, 403, 406, 411, 414, 417, 421, 425, 428, 433, 436, 439, 442, 445, 448, 451, 454, 458, 460, 463, 467
Output parameters .....	231, 234, 245, 248, 250, 256, 259, 261, 262, 267, 271, 273, 276, 285, 289, 291, 293, 295, 297, 310, 312, 355, 359, 363, 367, 372, 377, 392, 397, 430
Output type OUT PWM-n-A .....	56
Output type OUT PWM-n-B .....	57
Output type OUT PWM-n-BRIDGE-A .....	57
Output type OUT Supply-A .....	58
Output type OUT Voltage-A .....	59
Output types .....	56
Output, 1-channel, safe .....	174
Output, 2-channel, safe, without output group .....	176
OutputGroup .....	317
Outputs (technology) .....	55
Overview .....	
documentation for CODESYS 3.n .....	9
Hardware .....	42
Project structure with CR7xxS .....	78
Software .....	64
User documentation for CR7xxS .....	9
Overview of operating mode states .....	207
Overview of user rights .....	495

## P

Period .....	275
Preparation of the PLC .....	83
Process image .....	507
Programming of the safety application .....	135
Programming of the standard application .....	114
Purpose of the document .....	7
PWM .....	508
PWM1000 .....	332

## R

ratiometric .....	508
RawCAN .....	
Control CAN network nodes .....	129
Request and send remote CAN messages .....	129
Send and receive CAN messages .....	129
Read device information .....	124
Read the device information .....	199
Required previous knowledge .....	15
Reset .....	213, 226
Reset application (cold) .....	214
Reset application (origin) .....	214
Reset application (warm) .....	214
Reset behaviour of the system .....	213
RESET_TYPE (ENUM) .....	227
Response time .....	40
Response to errors .....	505, 506

## S

Safe state .....	29
Safety architecture .....	25

Safety concept of the outputs .....	169
Safety data types .....	140
Safety instructions .....	15
Safety libraries .....	342
Safety programming .....	146
Safety requirements .....	138
Safety time .....	39
Safety-related applications .....	146
Self-test at restart (start-up test) .....	31
Serial interface .....	62
Set communication path of PLC .....	79
SetLED .....	291
Setting /measurement via FB Output .....	59
Setting the IP parameter of the Ethernet interface .....	101
Setting the task configuration .....	498
Set-up and maintenance .....	187
SF_Antivalent .....	411
SF_Antivalent_BOOL .....	394
SF_CamshaftMonitor .....	413
SF_CurrentControl .....	382
SF_CurrentControlEnh .....	365
SF_DoubleValveMonitoring .....	416
SF_EDM .....	420
SF_EmergencyStop .....	424
SF_EnableSwitch .....	427
SF_Equivalent .....	430
SF_Equivalent_BOOL .....	392
SF_Equivalent_DINT .....	396
SF_Equivalent_REAL .....	405
SF_Equivalent_UDINT .....	402
SF_Equivalent_UINT .....	399
SF_ESPE .....	432
SF_FootSwitch .....	435
SF_GuardLocking .....	438
SF_GuardMonitoring .....	441
SF_HBridge .....	388
SF_HBridgeEnh .....	370
SF_Input .....	344
SF_InputBlanking .....	349
SF_ModeSelector .....	444
SF_OutControl .....	447
SF_OutGroupEnh .....	357
SF_Output .....	376
SF_OutputEnh .....	353
SF_OutputGroup .....	379
SF_PWM1000 .....	385
SF_PWM1000Enh .....	361
SF_SafetyRequest .....	450
SF_SingleValveCycleMonitoring .....	453
SF_SingleValveMonitoring .....	457
SF_TwoHandControlTypell .....	460
SF_TwoHandControlTypelll .....	463
SF_ValveGroupControl .....	466
Sign the user in to the CODESYS project .....	85
Sleep mode .....	48
Software .....	70
Software description .....	64
Software in the controller .....	64
Software information (swinfo.txt) .....	201
Software module for the device .....	65
Software on the PC/notebook .....	64
Stack .....	99
Standard diagnostic limit values of inputs .....	121
Standard diagnostic limit values of outputs .....	122
Standard PLC and safety PLC .....	23
Start CODESYS .....	75
Start conditions .....	46
Start-up behaviour of the controller .....	16
State chart .....	348, 352
State chart SF_Equivalent_[Typ] .....	408
State diagram SF_[Type]Enh .....	374
Status LED .....	
Ethernet interfaces (ETH0, ETH1) .....	212
Sleep mode (SYS0) .....	211
System bootloader (SYS0) .....	211
system ifm operating system (SYS0+SYS1) .....	210
system PLC (SYS0, SYS1) .....	210
Status LEDs .....	210
SupplySwitch .....	293
Supported programming languages .....	116, 138
Supported variable types .....	118, 140
Switch off outputs via solid-state switch .....	48
Switch on/off via ignition lock (clamp 15) .....	46
Switch on/off via main switch .....	46
Switching off an output group safely .....	178
Symbols used .....	8
SYS_INFO (STRUCT) .....	340
SYS_VOLTAGE_CHANNEL (ENUM) .....	306
SysInfo (GVL) .....	222
SysInfoStruct (STRUCT) .....	223
System (in general) .....	23
System architecture .....	25
System configuration .....	82
System context of the controller .....	23
System description .....	22
System overview .....	42
System requirements .....	70
SystemSupply .....	295
<b>T</b> .....	
Task configuration example .....	497
Task monitoring .....	504
Temperature .....	297
The process safety time .....	33
Time response .....	32
Time-related behaviour IEC application .....	36
Time-related behaviour of the inputs .....	35
Time-related behaviour of the outputs .....	37
Transfer CODESYS project to device .....	191
Transmission of the files with CODESYS .....	195
Troubleshooting .....	471
Types of inputs .....	51
<b>U</b> .....	
Undetected errors (latent errors) .....	31
Uninstall package (PC/laptop) .....	73
Update package (PC/laptop) .....	72
Update the operating system of the device .....	189
Update the operating system of the device with the batch file .....	189
Update the operating system of the device with the ifm Maintenance Tool .....	189
Usage in applications according to ISO 13849-1 .....	27

Use CANopen .....	130
Use CANopen-Safety .....	182
Use IO mapping .....	125
Use memory .....	97
Use RawCAN (CAN Layer 2) .....	129
Use SAE J1939 .....	131
Use the CODESYS operating instructions .....	75
User-defined data .....	89
Using Ethernet .....	132
Using ifm function libraries .....	123
Using libraries that are and that are not from ifm .....	136
Using the function blocks .....	216
Using the operators .....	486

## V

via function block .....	111
RAW-CAN .....	110
via safety function block .....	111
via system configuration .....	112
CANopen device .....	105
CANopen Manager .....	103
CANopen manager SIL2 .....	106
CANopen-Device SIL2 .....	107
J1939 manager .....	108
Voltage ranges of the on-board system .....	45

## W

Warnings used .....	8
Watchdog .....	30