



Programming manual

**ecomatDisplay**

Operating system: V2.x.x.x or higher  
CODESYS version: 3.5 SP16 Patch 0

**GB**

---

# Contents

1	Preliminary note	6
1.1	Legal and copyright information	6
1.2	Purpose of the document	6
1.3	Symbols used	7
1.4	Warnings used	7
1.5	Overview: ifm user documentation	7
1.6	Overview: CODESYS documentation	8
1.7	Change history	8
2	Safety instructions	10
2.1	Required background knowledge	10
2.2	Cyber security	10
3	Installation	11
3.1	System requirements	11
3.1.1	Hardware	11
3.1.2	Software	11
3.1.3	Licensing	11
3.2	CODESYS Development System	11
3.2.1	Installing CODESYS Development System	11
3.3	ifm package	12
3.3.1	Components of the package	12
3.3.2	Installing the package	12
3.3.3	Updating the package	12
3.3.4	Uninstalling the package	13
3.4	Updating the runtime system of the device	13
3.4.1	General notes	13
3.4.2	Starting the recovery mode	13
3.4.3	Updating the runtime system	14
3.4.4	The the IP parameter of the Ethernet interface	16
3.4.5	Quitting the recovery mode	17
4	Getting started	18
4.1	Start CODESYS	18
4.2	Creating a CODESYS project	18
4.2.1	Template for ecomatDisplay	18
4.2.2	Overview: Project structure with ecomatDisplay	18
4.2.3	Creating a new project with ecomatDisplay	19
4.3	Using the CODESYS operating instructions	19
4.4	Configuring the programming interface	19
4.4.1	Setting the communication path of the PLC	20
4.5	Activating the access protection for a project	21
4.6	Accessing the Linux system of the device	21
5	Device set-up	22
5.1	Starting the set-up mode	22
5.1.1	Main menu setup: Sub-menus	23
5.1.2	Notes on operation	23
5.1.3	Connection	26
5.1.4	Backup: Creating a data backup	27
5.1.5	System setup	28
5.1.6	Device Diagnostic	33
5.1.7	Device Info: show device information	36
5.1.8	Removing USB: Removing the USB stick safely	36
5.2	Starting the PLC application	37
5.3	Quitting the setup, rebooting the device	37
6	System configuration	38
6.1	Configuring the PLC	38
6.2	Adding a font	38
6.3	Configuring CAN interfaces	38
6.3.1	Device description files (EDS files)	39

6.3.2	Adding and configuring a CANbus .....	39
6.3.3	RawCAN: configuring CANLayer 2 .....	39
6.3.4	CANopen: configuring CANopen Manager (master) .....	40
6.3.5	CANopen: configuring the CANopen Device (slave) .....	40
6.3.6	J1939: configuring J1939 Manager .....	41
7	Programming .....	42
7.1	Objects of the PLC application with ecomatDisplay template .....	42
7.2	Creating a PLC application .....	43
7.2.1	Supported programming languages .....	43
7.2.2	PLC_PRG in FUP and ST .....	43
7.2.3	Available memory .....	44
7.2.4	Supported variable types .....	44
7.2.5	Persistent variables .....	44
7.2.6	Symbol names of the operating elements .....	45
7.2.7	Procedure .....	46
7.3	Using ifm function libraries .....	46
7.3.1	Configuring the device .....	46
7.3.2	Controlling the device .....	46
7.3.3	Executing and configuring audio functions .....	47
7.3.4	Configuring the Ethernet interface .....	47
7.3.5	Configuring device keys .....	47
7.3.6	Configuring the device display .....	48
7.3.7	Accessing device sensors and inputs/outputs .....	48
7.3.8	Configuring/reading system time .....	48
7.3.9	File management .....	49
7.3.10	Configuring the touch screen .....	49
7.3.11	Setting and controlling the analogue camera .....	49
7.3.12	Setting and controlling the Ethernet camera .....	49
7.3.13	Configuring the PDF viewer .....	49
7.3.14	Controlling image fields / making a screenshot .....	49
7.3.15	Managing CSV files .....	50
7.3.16	Using help functions .....	50
7.4	Using visualisations .....	51
7.4.1	Settings in the project template .....	51
7.4.2	Integrating external files .....	51
7.4.3	Texts and fonts .....	51
7.4.4	Language selection .....	52
7.4.5	Using image pools .....	53
7.4.6	Using the visualisation manager .....	54
7.4.7	Creating a visualisation .....	57
7.5	Using touch screen functions .....	57
7.5.1	Notes .....	57
7.5.2	Configuring input objects of the visualisation .....	58
7.5.3	Using multitouch functionality .....	58
7.6	Operation without touch functionality .....	58
7.7	Using mobile cameras .....	59
7.7.1	Supported cameras .....	59
7.7.2	Configuring and controlling the analogue camera .....	59
7.7.3	Configuring and controlling an Ethernet camera .....	60
7.7.4	Configuring the Region of Interest (ROI) .....	60
7.8	Using a PDF viewer .....	61
7.8.1	Example .....	61
7.9	CSV file logging .....	62
7.9.1	Writing a CSV file .....	62
7.9.2	Reading a CSV file .....	63
7.9.3	String format uiGenericLogSizeMax .....	63
7.10	CODESYS IIoT Libraries SL .....	63
7.11	Using CANopen .....	64
7.11.1	CANopen: Sending and receiving SDO .....	64
7.11.2	CANopen: Network Management (NMT) .....	64
7.12	Using RawCAN (CAN Layer 2) .....	64

7.12.1	RawCAN: Controlling CAN network nodes	64
7.12.2	RawCAN: Sending and receiving CAN messages	64
7.12.3	RawCAN: Requesting and sending remote CAN messages	65
7.13	Using J1939	65
7.13.1	Adding a CAN bus	65
7.13.2	Assign CAN interface	65
7.13.3	Adding J1939 Manager	65
7.13.4	Setting J1939 Manager parameters	66
7.13.5	Attaching J1939-ECU	66
7.13.6	Setting J1939-ECU parameters	66
7.14	Using EtherNet/IP	67
7.14.1	Adding an Ethernet adapter to an Ethernet	67
7.14.2	Attaching an EtherNet/IP adapter	67
7.14.3	Attaching an EtherNet/IP module	67
7.14.4	Configuring the EtherNet/IP interface	68
7.15	Using Modbus	68
7.15.1	Adding an Ethernet adapter to an Ethernet	68
7.15.2	Attaching a Modbus TCP master	68
7.15.3	Attaching a Modbus TCP slave device	68
7.15.4	Configuring a Modbus TCP slave device	69
7.16	Configuring task processing	69
7.16.1	Configuring a task	70
7.16.2	Configuring a visualisation task	70
8	Operation	71
8.1	Transferring CODESYS project to the device	71
8.1.1	Loading an application to ecomatDisplay	71
8.1.2	Deleting an application from the device	71
8.2	Operating states of the PLC application	72
8.2.1	Displaying operating mode of the PLC application	72
8.2.2	Starting the PLC application	72
8.2.3	Stopping the PLC application	72
8.3	Reset	72
8.3.1	Supported reset variants	72
8.3.2	Resetting the application (warm)	73
8.3.3	Resetting the application (cold)	73
8.3.4	Resetting the application (origin)	73
8.4	Displaying system information	74
9	ifm function libraries	75
9.1	Behaviour model of the ifm function blocks	75
9.1.1	General	75
9.1.2	Behaviour model ENABLE	75
9.1.3	Behaviour model EXECUTE	76
9.2	Library ifm_ecomatDisplay_Cnt	77
9.3	ifmCANOpenManager.library	78
9.3.1	COP_GetNodeState	78
9.3.2	COP_SDRead	80
9.3.3	COP_SDWrite	82
9.3.4	COP_SendNMT	84
9.3.5	NMT_SERVICE (ENUM)	85
9.3.6	NMT_STATES (ENUM)	85
9.4	Library ifmDevice_ecomatDisplay.library	86
9.4.1	Audio	86
9.4.2	Common	94
9.4.3	Ethernet	108
9.4.4	Keypads	118
9.4.5	LCD	132
9.4.6	Local IO	142
9.4.7	Status LED	159
9.4.8	Storage	161
9.4.9	System Commands	165
9.4.10	System Information	173

---

9.4.11	Touch	175
9.4.12	System Time	179
9.4.13	Window Control	193
9.4.14	ENUM	204
9.4.15	STRUCT	208
9.4.16	Global	219
9.5	ifmFileUtil.library	220
9.5.1	Generic File	220
9.5.2	Parameter File	230
9.5.3	Log File	234
9.5.4	Support	251
9.5.5	Function	257
9.5.6	ENUM	268
9.5.7	STRUCT	268
9.5.8	GlobalConstants	269
9.6	ifmRawCAN.library	271
9.6.1	CAN_Enable	271
9.6.2	CAN_Recover	273
9.6.3	CAN_RemoteRequest	275
9.6.4	CAN_RemoteResponse	277
9.6.5	CAN_Rx	279
9.6.6	CAN_RxMask	281
9.6.7	CAN_RxRange	283
9.6.8	CAN_RxRangeExt	285
9.6.9	CAN_Status	287
9.6.10	CAN_Tx	289
9.6.11	BUS_STATE (ENUM)	291
9.6.12	CAN_Info (GVL)	291
9.6.13	CAN_BUS_STATE (STRUCT)	291
10	Appendix	292
10.1	Address assignment in Ethernet networks	292

---

# 1 Preliminary note

You will find instructions, technical data, approvals and further information using the QR code on the unit / packaging or at [www.ifm.com](http://www.ifm.com).

## 1.1 Legal and copyright information

© All rights reserved by ifm electronic gmbh. No part of these instructions 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 AS-International Association, (→ [www.as-interface.net](http://www.as-interface.net))
- CAN is the property of Robert Bosch GmbH, Germany (→ [www.bosch.de](http://www.bosch.de))
- CAN is the property of CiA (CAN in Automation e.V.), Germany (→ [www.can-cia.org](http://www.can-cia.org))
- CODESYS™ is the property of CODESYS GmbH, Germany (→ [www.codesys.com](http://www.codesys.com))
- DeviceNet™ is the property of ODVA™ (Open DeviceNet Vendor Association), USA (→ [www.odva.org](http://www.odva.org))
- EtherNet/IP® is the property of → ODVA™
- EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
- IO-Link® is the property of PROFIBUS Nutzerorganisation e.V., Germany (→ [www.io-link.com](http://www.io-link.com))
- ISOBUS is the property of AEF - Agricultural Industry Electronics Foundation e.V., Germany (→ [www.aef-online.org](http://www.aef-online.org))
- Microsoft® is the property of Microsoft Corporation, USA (→ [www.microsoft.com](http://www.microsoft.com))
- Modbus® is the property of Schneider Electric SE, France (→ [www.schneider-electric.com](http://www.schneider-electric.com))
- PROFIBUS® is the property of PROFIBUS Nutzerorganisation e.V., Germany (→ [www.profibus.com](http://www.profibus.com))
- PROFINET® is the property of → PROFIBUS Nutzerorganisation e.V., Deutschland
- Windows® is the property of → Microsoft Corporation, USA

## 1.2 Purpose of the document



This document applies to the following devices of the type ecomatDisplay firmware version V 2.x.x.x and higher:

- CR1058
- CR1059
- CR1074
- CR1075
- CR1076
- CR1077
- CR1102
- CR1202
- CR1203
- CR1204




These instructions describe the following topics:

- Configuration of the device in the setup mode
- Firmware update of the device in the recovery mode
- Configuration of the device with CODESYS 3.5
- Programming of the device-internal PLC using the CODESYS 3.5 programming system.
- Description of the device-specific CODESYS function libraries

## 1.3 Symbols used

- ✓ Requirement
- Instructions
- ▷ Reaction, result
- [...] Designation of keys, buttons or indications
- Cross-reference
-  Important note  
Non-compliance may result in malfunction or interference.
-  Information  
Supplementary note

## 1.4 Warnings used

	<b>ATTENTION</b> Warning of damage to property
	<b>CAUTION</b> Warning of personal injury ▷ Slight reversible injuries may result.
	<b>WARNING</b> Warning of serious personal injury ▷ Death or serious irreversible injuries may result.

## 1.5 Overview: ifm user documentation

The documentation of the device consists of the following modules:

Document	Content / Description
Data sheet	<ul style="list-style-type: none"> <li>• Technical data</li> </ul>
Installation instructions / 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>

---

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: CODESYS documentation

CODESYS GmbH provides the following user documentation for programming the PLC of the device:

Document	Content / description
Online help	<ul style="list-style-type: none"><li>• Context-sensitive help</li><li>• Description of the CODESYS programming system</li><li>• Description of the components and libraries</li></ul>
CODESYS installation and first steps	<ul style="list-style-type: none"><li>• Remarks about the installing of the programming system CODESYS</li><li>• First steps for handling the CODESYS programming system</li></ul>

After the installation of the programming system CODESYS 3.5 all documents are stored on the hard disk of the PC/laptop and can be accessed:

- Online help  
... \Programs (x86) \3S CODESYS \CODESYS \Online Help
- CODESYS installation and first steps  
... \Programs (x86) \3S CODESYS \CODESYS \Documentation

## 1.7 Change history

Output	Topic	Date
00	New creation of the document	11/2019



Output	Topic	Date
01	<p>Changes for Firmware V2</p> <p><b>New chapters:</b></p> <p>Persistent variables (→ <a href="#">44</a>)</p> <p>Configuring and controlling an Ethernet camera (→ <a href="#">60</a>)</p> <p>Using a PDF viewer (→ <a href="#">61</a>)</p> <p>CSV file logging (→ <a href="#">62</a>)</p> <p>CODESYS IIoT Libraries SL (→ <a href="#">63</a>)</p> <p>Using J1939 (→ <a href="#">65</a>)</p> <p>Using EtherNet/IP (→ <a href="#">67</a>)</p> <p>Using Modbus (→ <a href="#">68</a>)</p> <p>Setting and controlling the Ethernet camera (→ <a href="#">49</a>)</p> <p>Configuring the PDF viewer (→ <a href="#">49</a>)</p> <p>Language selection (→ <a href="#">52</a>)</p> <p>Operation without touch functionality (→ <a href="#">58</a>)</p> <p>Global switching of the visualisation (→ <a href="#">55</a>)</p> <p>Local switching of the visualisation (→ <a href="#">55</a>)</p> <p>Configuring the target visualisation (→ <a href="#">56</a>)</p> <p>Configure web visualisation (→ <a href="#">56</a>)</p> <p>File name conventions (→ <a href="#">51</a>)</p> <p><b>Revised chapters:</b></p> <p>Purpose of the document (→ <a href="#">6</a>)</p> <p>Adjusting the boot screen (→ <a href="#">31</a>)</p> <p>Managing CSV files (→ <a href="#">50</a>)</p> <p>Using help functions (→ <a href="#">50</a>)</p> <p>Using mobile cameras (→ <a href="#">59</a>)</p> <p>Supported cameras (→ <a href="#">59</a>)</p> <p>Settings in the project template (→ <a href="#">51</a>)</p> <p>Integrating external files (→ <a href="#">51</a>)</p> <p><b>Removed chapters:</b></p> <p>Description eKEY_ID</p> <p><b>New library descriptions:</b></p> <ul style="list-style-type: none"> <li>• FB PDF_Viewer</li> <li>• FB IPCameraWindowControl</li> <li>• ENUM eIP_CAMERA_ENCODING</li> <li>• ENUM eIP_CAMERA_PROTOCOL</li> <li>• STRUCT stCAMERA_CONFIG</li> <li>• STRUCT stPDF_CONTROLS</li> <li>• ALIAS aETH_ITF_LIST</li> <li>• ALIAS aUSB_STORAGE_INFO_LIST</li> <li>• FB ReadCSVData</li> <li>• FB WriteCSVData_Linear</li> <li>• FB WriteCSVData_Ring</li> <li>• WriteCSVHeader</li> <li>• FUN ANY_TYPE_TO_STRING</li> <li>• FUN ifmCONCAT</li> <li>• FUN ifmFIND</li> <li>• FUN ifmMID</li> <li>• STRUCT stLOG_FILE_CONFIG</li> <li>• STRUCT stSEPARATOR_CONFIG</li> <li>• GVL ifmGCL</li> </ul>	01/2021

---

## 2 Safety instructions

- The unit described is a subcomponent for integration into a system.
  - The system architect is responsible for the safety of the system.
  - The system architect undertakes to perform a risk assessment and to create documentation in accordance with legal and normative requirements to be provided to the operator and user of the system. This documentation must contain all necessary information and safety instructions for the operator, the user and, if applicable, for any service personnel authorised by the architect of the system.
- Read this document before setting up the product and keep it during the entire service life.
- The product must be suitable for the corresponding applications and environmental conditions without any restrictions.
- Only use the product for its intended purpose (→ Installation instructions / Operating instructions, Intended use).
- If the operating instructions or the technical data are not adhered to, personal injury and/or damage to property may occur.
- The manufacturer assumes no liability or warranty for any consequences caused by tampering with the product or incorrect use by the operator.
- Improper or non-intended use may lead to malfunctions of the device, to unwanted effects in the application or to a loss of the warranty claims.
- Installation, electrical connection, set-up, programming, configuration, operation and maintenance of the product must be carried out by personnel qualified and authorised for the respective activity.
- Observe applicable documents.

### 2.1 Required background knowledge

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.

For programming, they should also have knowledge of control technology experience in PLC programming to IEC 61131-3.

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

### 2.2 Cyber security

---

#### ATTENTION

Operating the machine in an unprotected network environment

- ▷ Unauthorised read or write access to data is possible.
  - ▷ Unauthorised manipulation of the device function is possible.
  - ▶ Check and restrict access options to the device.
-

## 3 Installation

### 3.1 System requirements

#### 3.1.1 Hardware

The following hardware components are required to program the internal PLC of the ecomatDisplay:

- A device of the ecomatDisplay product family
- A PC for the CODESYS programming system
- An Ethernet connection between CODESYS PC and the Ethernet interface of the unit.

#### 3.1.2 Software

To program the device-internal PLC of the ecomatDisplay, the following software components are required:

Component	Description	Version
CODESYS Development System	Programming software CODESYS for PLC programming according to standard IEC 61131-3	3.5 SP16 Patch 0
Package "CODESYS for ifm ecomatDisplay"	<ul style="list-style-type: none"> <li>• Device and interface description of device</li> <li>• Function libraries for the programming of the device</li> </ul>	2.x.x.x



The features and functions warranted in this manual can only be achieved with the software components in the versions specified here.

The software components can be downloaded on the ifm electronic website: [www.ifm.com](http://www.ifm.com)

#### 3.1.3 Licensing

By purchasing the ecomatDisplay, the user also acquires a valid licence to use the "CODESYS for ifm ecomatDisplay".



► Licence information → Installation routine or product page of the article on the ifm website.

## 3.2 CODESYS Development System

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

### 3.2.1 Installing CODESYS Development System

To install the software "CODESYS Development System":

- Install the CODESYS programming system 3.5 SP16 Patch 0
- ▷ CODESYS 3.5 SP16 Patch 0 is installed on the PC/laptop.

---

## 3.3 ifm package



- Familiarise yourself with the following CODESYS functions:  
Package Manager: → Online help > CODESYS Development System > Manage packages and licenses

### 3.3.1 Components of the package

ifm provides the CODESYS package "CODESYS for ifm ecomatDisplay" for the programming of the device-internal PLC. The package contains the following components:

- Device description files
- ifm function libraries and extensions



- Detailed information about the ifm function libraries: ifm function libraries (→ [75](#))

### 3.3.2 Installing the package

To install the ifm package "CODESYS for ifm ecomatDisplay":

- ✓ CODESYS3.5 SP16 Patch 0 has been correctly installed.
- ✓ ifm package "CODESYS for ifm ecomatDisplay" is stored on the PC/laptop.
- Start CODESYS with administrator rights.
  - ▷ CODESYS starts.
  - ▷ CODESYS user interface appears.
- Select [Tools] > [Package Manager].
  - ▷ The [Package Manager] window appears.
- Click on [Install...]
  - ▷ The file explorer is displayed.
- Select the file ifm\_ecoSys\_Lx\_64bit\_Vx.x.x.x.package
- Click on [Open] and execute a complete installation.
  - ▷ "CODESYS for ifm ecomatDisplay" is installed.
  - ▷ After successful installation: The [Package Manager] window shows the installed package.
  - ▷ The functions of the package can be used.
- Click on [Close] to quit the Package Manager.

### 3.3.3 Updating the package

To update the ifm package "CODESYS for ifm ecomatDisplay":

- Uninstall the old version of the ifm package: Uninstalling the package (→ [13](#))
- Install the new version of the ifm package: Installing the package (→ [12](#))
- Open the project.
- In the device tree: Select the node of the device.
- 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 close the dialogue window.
- ▶ Save the project.

### 3.3.4 Uninstalling the package

To install the ifm package "CODESYS for ifm ecomatDisplay":

- ▶ Start the Package Manager: Select [Tools] > [Package Manager].
  - ▷ The [Package Manager] window shows the installed packages.
- ▶ Select the package to be uninstalled.
- ▶ Click on [Uninstall..].
  - ▷ The selected package will be uninstalled.
- ▶ Click on [Close] to quit the Package Manager.

## 3.4 Updating the runtime system of the device

### 3.4.1 General notes

To update the runtime system, the unit must be in recovery mode.

#### Downloading the update file

- ▶ Download the current version of the runtime system from the ifm website: [www.ifm.com](http://www.ifm.com) > Product page > [Downloads] > [Software download]
- ▶ Unpack ZIP archive with the update file.
- ▷ The update file is stored on the PC.

### 3.4.2 Starting the recovery mode






- ▶ Separate the device from the circuit.
- ▶ Perform the following actions **simultaneously**:
  - Connect the SERVICE1 terminal to VBB.
  - Connect the SERVICE0 terminal to GND.
  - Switch the device on again.
  - To do so, both terminal 15 and terminal 30 must be connected to VBB.
- ▷ The device reboots.
- ▷ Device is in recovery mode.

#### Alternative option for units with integrated key panel:

- ▶ Separate the device from the circuit.
- ▶ Perform the following actions **simultaneously**:
  - Press 3 random keys on the unit simultaneously and keep them pressed.
  - Switch the device on again.
  - To do so, both terminal 15 and terminal 30 must be connected to VBB.
- ▷ The device reboots.
- ▶ Keep the keys pressed until the ifm logo appears / status LED flashes orange.
- ▷ Device is in recovery mode.

## Notes on operation

In the recovery mode, the unit is operated using the touch screen or with the navigation keys:

Navigation keys	Key	Function
	[▲]	Move up Select previous menu element
	[◀]	Move to the left select previous element
	[▶]	Move to the right select next element
	[▼]	Move down select next menu element
	[RETURN]	Selecting a menu item Enabling a button Increasing the value

## Menu in recovery mode

The screen in the recovery mode shows:

- Following sub-menus:

Button	Meaning
[UPDATE FROM FILE]	Runtime system / firmware update via USB interface (→ 14)
[NETWORK SETUP]	The the IP parameter of the Ethernet interface (→ 16)
[RESTART]	Reboot the device.

- Current parameter settings of the Ethernet interface

## 3.4.3 Updating the runtime system

### ATTENTION

Risk of data loss!

- ▷ Interrupting the update process may result in a loss of the user data stored on the device.
- ▶ Do not interrupt the update process!



The runtime system of the device can only be updated in the recovery mode.

The current version of the runtime system can be downloaded from the ifm electronic website.  
Downloading the update file (→ 13)

- ▶ Select one of the following options:
- Runtime system / firmware update via USB interface (→ 14)
- Runtime system / firmware update via web browser (→ 15)

### Runtime system / firmware update via USB interface

- ▶ Download the new runtime system from the ifm website: Downloading the update file (→ 13)

- ▶ Copy the update file \*.swu to a USB memory.
- ▶ Connect the USB memory with the device: → Installation instructions
- ▶ Starting the recovery mode: Starting the recovery mode (→ 13)



- ▶ Do not press any buttons on the device during the update process.
- ▶ Do not interrupt the voltage supply during the update process.

- ▶ Open the file browser with the [INSTALL FROM FILE] button.
  - ▷ The file browser appears.
- ▶ Select the update file \*.swu in the file browser.
- ▶ Start the update process with the [OPEN] button.
  - ▷ Software components of the device are updated automatically: General notes (→ 13)
  - ▷ The display shows status messages
  - ▷ If successful: status message appears on the display: [Success!]
- ▶ Reboot the device using the [RESTART] button.
  - ▷ The device reboots.
  - ▷ The firmware update is finished.

### Runtime system / firmware update via web browser

- ▶ Download the new runtime system from the ifm website: Downloading the update file (→ 13)
- ▶ Starting the recovery mode: Starting the recovery mode (→ 13)
- ▶ Establish an Ethernet network connection between the device and the PC: → Installation instructions
- ▶ Optional: Set the parameters of the Ethernet interface.
- ▶ Start the web browser on the PC.
- ▶ Enter the following into the address line of the browser: `http://<IP address of the device>:8080`; Default: `http://192.168.82.247:8080`
- ▶ Confirm by pressing the enter button.
  - ▷ The web interface of the device appears in the browser.
  - ▷ The web interface immediately changes to the [Software Update] tab.



- ▶ Do not press any buttons on the device during the update process.
- ▶ Do not push the [Restart] button in the web interface during the update process.
- ▶ Do not interrupt the voltage supply during the update process.

- ▶ Click into the file area [Click here...].
  - ▷ The Windows file explorer is displayed.
- ▶ Select the downloaded update file \*.swu and adopt it with [Open] and start the update.
  - ▷ The file area shows the file name of the selected update file.
  - ▷ The selected update file is loaded to the display.
  - ▷ Software components of the device are updated automatically: General notes (→ 13)
  - ▷ The web interface and the device display indicate status messages.

- 
- ▷ If successful: status message appears in the web interface: [Update successfully] .
  - ▶ Click the [Restart] button in the web interface.
  - ▷ The device reboots.
  - ▷ The firmware update is finished.

### 3.4.4 The the IP parameter of the Ethernet interface

In order to update the runtime system of the device via a network, the device must be connected to the corresponding network.



Standard settings of the IP address:

IP address = 192.168.82.247

Subnet mask = 255.255.255.0

Gateway address = 192.168.82.1

For the configuration of the Ethernet interface, the following options are available:

- Manually: The operator sets the interface parameters (IP address, subnet mask, gateway address) manually.
- Automatic: The interface parameters are set via the Dynamic Host Configuration Protocol (DHCP).

#### ATTENTION

If the device is operated in an unprotected network environment.

- ▷ Unauthorised read or write access to data is possible.
- ▷ Unauthorised manipulation of the device function is possible.
- ▶ Check and restrict access options to the device.
- ▶ Restrict access to authorised users.
- ▶ Choose a secure method to connect with the device (e.g. VPN).
- ▶ Use encrypted data transmission (e.g. https / TLS).

- ▶ To configure the IP parameters of the Ethernet interface:
- ▶ Select one of the following options:
  - Obtaining the IP parameters from a DHCP server (→ 16)
  - Configuring the IP parameters manually (→ 17)

#### Obtaining the IP parameters from a DHCP server

- ▶ Connect the Ethernet interface of the device with the IP network.
- ▶ Start the device in the recovery mode: Starting the recovery mode (→ 13)
- ▶ Select the [NETWORK SETUP] using [▲] / [▼] and enable with [RETURN].
  - ▷ The menu page shows current settings ([DHCP], [IP-ADDRESS], [NETMASK]) of the existing network interfaces [eth0], [eth1] and [Gateway].
- ▶ Use [▲] / [▼] to select the network interface and confirm with [RETURN].
  - ▷ The selected network interface is highlighted.
- ▶ Use [◀] / [▶] to select the control field [DHCP] and enable with mit [RETURN].
- ▶ Use [◀] / [▶] to select the [APPLY] button and enable with [RETURN].



- ▷ DHCP is enabled for the selected network interface. In the [DHCP] column, [yes] is shown.
- ▷ The device tries to receive IP parameters of the Ethernet interface from the DHCP server of the network.
- ▷ If successful:
- ▷ the [NETWORK SETUP] menu page appears.
- ▷ The information line indicates current values of the IP parameters [DHCP], [IP-ADDRESS], [NETMASK].
- ▷ The device is available under the displayed parameters in the IP network.
- ▶ Optional: To check the availability of the device, execute the following command in the prompt: `ping <device address>`, e.g. `ping 192.168.82.247`

### Configuring the IP parameters manually



- ▶ Further infos about the assignment of IP addresses in Ethernet networks: Address assignment in Ethernet networks (→ [292](#))
- ▶ Start the device in the recovery mode. Starting the recovery mode (→ [13](#))
- ▶ Select the [NETWORK SETUP] using [▲] / [▼] and enable with [RETURN].
  - ▷ The menu page shows current settings ([DHCP], [IP-ADDRESS], [NETMASK]) of the existing network interfaces [eth0], [eth1] and [Gateway].
- ▶ Use [▲] / [▼] to select the network interface and confirm with [RETURN].
  - ▷ The selected network interface is highlighted.
- ▶ Use [◀] / [▶] to select the first IP parameter in the [IP-Address] section.
  - ▷ The selected IP parameter is highlighted.
- ▶ Increment the required figure using [RETURN]. Keeping [RETURN] pressed = fast incrementation of the figure.
  - ▷ The input field shows the set value.
- ▶ Use [◀] / [▶] to select the next input field and to set the required value.
- ▶ Repeat the process until the required IP address is set.
- ▶ Repeat the steps described above for the parameters of the subnet mask in the [Netmask] section.
- ▶ Use [◀] / [▶] to select the [APPLY] button and enable with [RETURN].
  - ▷ The set parameter values are enabled.
  - ▷ If successful:
  - ▷ the [NETWORK SETUP] menu page appears.
  - ▷ The information line indicates current values of the IP parameters [DHCP], [IP-ADDRESS], [NETMASK].
  - ▷ The device is available under the displayed parameters in the IP network.
- ▶ Optional: To check the availability of the device, execute the following command in the prompt: `ping <device address>`, e.g. `ping 192.168.82.247`

### 3.4.5 Quitting the recovery mode

- ▶ Call up the recovery screen.
- ▶ Use [▲] / [▼] to select the menu item [RESTART] and confirm with [RETURN].
- ▷ The device reboots.



- ▶ If a valid project is loaded on the device, this project will be started automatically after a reboot.
- ▶ If no valid project has been stored, the set-up menu will appear after rebooting

## 4 Getting started

### 4.1 Start CODESYS.

- ✓ Software components are correctly installed. Installation (→ 11)
- ▶ Start CODESYS. Double click on the icon [CODESYS 3.5 SP16 Patch 0]
- ▷ CODESYS starts.
- ▷ CODESYS user interface appears.

### 4.2 Creating a CODESYS project



- ▶ Familiarise yourself with the following CODESYS functions:  
Creating a project → Online help > CODESYS Development System > Creating and Configuring a Project  
  
Managing a project: → Online help > CODESYS Development System > Protecting and Saving Projects

#### 4.2.1 Template for ecomatDisplay

ifm provides a special project template for each model of the device family. The template includes the optimum presets for the corresponding target device. The user can select the template during the creation of the project. The templates are available when the device package is installed.



- To avoid errors during manual system configuration, it is explicitly recommended to use the project template from ifm when creating the <ASi\_Line> project in CODESYS.

#### 4.2.2 Overview: Project structure with ecomatDisplay

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

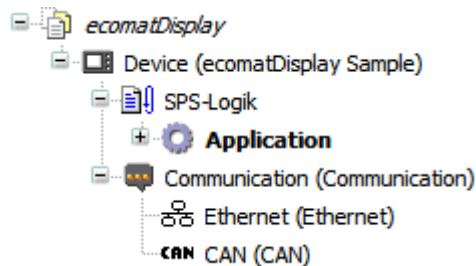


Fig. 1: Legend:

ecomatDisplay (ecomatDisplay sample)	Logical father controller, offers access to the general settings of the ecomatDisplay
PLC logic	Includes the application of the ecomatDisplay.
Application	Includes the objects that are required for a control program and the visualisation.
Communication	provides access to the configuration options of the communication interfaces

Ethernet / CAN	Available communication interfaces.
----------------	-------------------------------------

### 4.2.3 Creating a new project with ecomatDisplay

- ✓ All required software components have been correctly installed. Installation (→ 11)
- ✓ CODESYS successfully started.
- ▶ Select [File] > [New Project...] wählen.
  - ▷ The window [New Project] appears.
- ▶ Set the following values:
  - [Category]: select ecomatDisplay.
  - [Templates]: Select the proper template for the device: [ifm ecomatDisplay ...]
  - Name: Enter project name
  - City: Select the storage location of the project file.
- ▶ Click on [OK] to verify the entered values.
  - ▷ CODESYS creates a new project with ecomatDisplay.
  - ▷ The window [Devices] shows the device tree of the project. Overview: Project structure with ecomatDisplay (→ 18)
- ▶ Select [File] > [Save project].
- ▷ CODESYS saves the project.

## 4.3 Using the CODESYS operating instructions

This manual exclusively describes the integration, configuration and programming of the ecomatDisplay using the CODESYS programming system.

The CODESYS terminology is used to describe user actions and components of the user interface.

Standard CODESYS functions and mechanisms are not described. At the beginning of each section, there is a reference to the corresponding chapters of the CODESYS online help.

To access the CODESYS online help:

- ▶ Start CODESYS.
  - ▷ The CODESYS user interface appears.
- ▶ Press [F1].
  - ▷ The online help of the CODESYS programming system appears.



- ▶ Familiarise yourself with the CODESYS programming system! This applies in particular to the following topics:
  - Designations and functions of the elements of the user interface
  - Basic menu functions
  - Programming techniques and data management mechanisms
  - Fieldbus support

## 4.4 Configuring the programming interface

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



The device and PC can be coupled either directly or indirectly via an Ethernet network.

- ▶ Only use the recommended accessories for connection of the Ethernet interfaces! → Installation instructions
- ▶ For network connection, an experienced user or system administrator should set up the network addresses and the configuration.
- ▶ If necessary, configure the Ethernet interface of the device in the setup mode. Connection (→ [26](#))



Requirements for the connection between the CODESYS PC and the device:

- ✓ The firmware versions of the applied project template and the device are matching.
- ✓ The application on the device is open.
  - or –
- ▶ If no application is running on the device, click the [LOAD APPLICATION] button. Starting the PLC application (→ [37](#))

#### ATTENTION

If the device is operated in an unprotected network environment.

- ▷ Unauthorised read or write access to data is possible.
- ▷ Unauthorised manipulation of the device function is possible.
- ▶ Check and restrict access options to the device:
- ▶ Restrict access to authorised users.
- ▶ Choose a secure method to connect with the device (e.g. VPN).
- ▶ Use encrypted data transmission (e.g. https / TLS).

### 4.4.1 Setting the communication path of the PLC

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

- ✓ CODESYS PC/laptop and Ethernet interface of the device are connected.
- ✓ Optional: Adjust IP settings of the Ethernet interface.
- ▶ In the device tree: Double-click on the [Device (ecomatDisplay)] icon
- ▶ In the editor window: Select the [Communication] tab.
  - ▷ Editor window shows communication settings.
- ▶ Select the required gateway in the [Gateway] list.
  - ▷ List shows selected gateway.
- ▶ Enable [Scan Network...]
  - ▷ The [Select Device] appears.
- ▶ Select the gateway node and start the scan process with [Scan Devices].
  - ▷ CODESYS scans the network for devices.
  - ▷ Window shows the network path and detected devices.
- ▶ Select the node of the device and enable [OK] to set the communication path to the device-internal PLC.

▷ CODESYS can transfer data to the device-internal PLC.

## 4.5 Activating the access protection for a project



- ▶ Familiarise yourself with the following CODESYS functions:  
Protect and save the project: → Online help > CODESYS Development System > Protecting and Saving Projects

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

- ▶ Select [Project] > [Project Settings].
  - ▷ The [Project settings] window.
- ▶ Select [Security].
- ▶ Enable the checkbox [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 enabled. The project is encrypted.

## 4.6 Accessing the Linux system of the device

The user can access the Linux system of the device directly via the Ethernet interface. The following options are supported:

Option	Description	Example application
Telnet	Unencrypted access to the Linux command line	Putty (→ <a href="http://www.putty.org">www.putty.org</a> )
SSH	Encrypted access to the Linux command line	Putty (→ <a href="http://www.putty.org">www.putty.org</a> )
FTP	Unencrypted access to the Linux file system (copy files)	WinSCP (→ <a href="http://www.winscp.net">www.winscp.net</a> ) File transfer in CODESYS
SCP	Encrypted access to the Linux file system (copy files)	WinSCP (→ <a href="http://www.winscp.net">www.winscp.net</a> )



User name and password: Login data (→  29)

## 5 Device set-up

This chapter describes how to configure the device in setup mode.

### 5.1 Starting the set-up mode



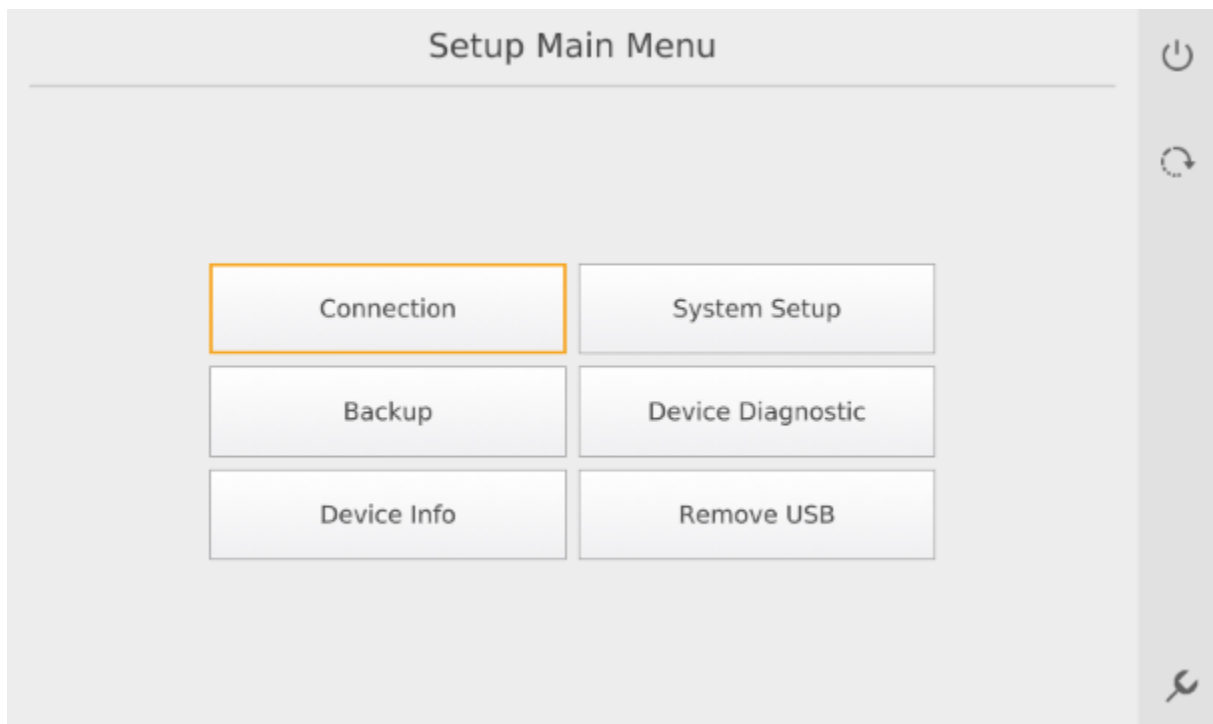
If no valid application is stored on the device, the device passes automatically to the start menu after power on. From there, the set-up mode can be started.

- ▶ Separate the device from the circuit.
- ▶ Perform the following actions **simultaneously**:
  - Connect the SERVICE0 terminal to VBB.
  - Switch the device on again. To do so, both terminal 15 and terminal 30 must be connected to VBB.
- ▷ The device reboots.
- ▶ Select the [Launch Setup] button in the menu and confirm with [RETURN].
- ▷ The device is in the set-up mode.

#### Alternative option for units with integrated key panel:

- ▶ Separate the device from the circuit.
- ▶ Perform the following actions **simultaneously**:
  - Press 2 random keys on the unit simultaneously and keep them pressed.
  - Switch on the device. To do so, both terminal 15 and terminal 30 must be connected to VBB.
- ▷ The device reboots.
- ▶ Select the [Launch Setup] button in the menu and confirm with [RETURN].
- ▷ The device is in the set-up mode.

#### Start page in the set-up mode:



### 5.1.1 Main menu setup: Sub-menus

The screen in the set-up menu [Setup Main Menu] shows the following sub-menus:

Button	Meaning
[Connection]	Settings of the Ethernet interfaces Connection (→ □ 26)
[System setup]	System settings System setup (→ □ 28)
[Backup]	Execute backup of the device data and settings Backup: Creating a data backup (→ □ 27)
[Device Diagnostic]	Device diagnostic Device Diagnostic (→ □ 33)
[Device Info:]	Device information Device Info: show device information (→ □ 36)
[Removing USB]	Removing the USB stick safely Removing USB: Removing the USB stick safely (→ □ 36)

### Rebooting the device

To reboot the device:

► Select the set-up main menu page.

► Reboot the device with .

▷ The device reboots.



If a valid project is loaded on the device, this project will be started automatically after a reboot.

If no valid project has been stored, the set-up menu will appear after rebooting






### 5.1.2 Notes on operation












In setup mode, the following rules apply to the control elements:

#### Key function

The function of the function keys changes depending on the context. The symbol on the corresponding button on the screen indicates which action will be triggered when a key is pressed.


The following table gives an overview of the available key functions:

Button	Meaning	Example
[▼], [▶]	Highlight next screen element	Select next menu item.
[▲], [◀]	Highlight previous screen element	Select previous menu item
[RETURN]	Enable function of the highlighted screen element	Call up sub-menu Change to the highlighted directory in the menu view
[▲], [▼], [◀], [▶]	<ul style="list-style-type: none"> <li>Navigate in the corresponding direction</li> <li>Increment / decrement the numerical value</li> </ul>	Select key on the on-screen keyboard
	Select the next higher menu level	Call up the next higher menu level
	Quit set-up.	-
	Next step:	-
	Restart the device.	-
	Playback.	Play audio text.

Button	Meaning	Example
	Starting the recovery mode.	-
	Open the file browser	-
	Remove USB stick safely.	-
	Save file or setting	Save IP address
	Start reception	Start reception of CAN messages
	Stop reception	Stop reception of CAN messages
	Start transmission	Start transmission of CAN messages
	Stop transmission	Stop transmission of CAN messages
	Show boot screen as full screen	-
	Show hidden characters.	Show the password.
	Confirmation / OK	Backup settings are ok. Start backup process.



### Operation: navigating in the setup menu

To navigate in the setup menu:

- ▶ Use [▼] or [▶] to select the next menu item (forward).
  - or -
  - Use [▲] or [◀] to select the previous menu item (backward).
  - ▷ Selected menu item has focus (orange frame).
- ▶ Change to the selected menu page with [RETURN].
  - or -
  - Change to the higher level menu page with .

### Operation: navigating on a menu page

To navigate within the menu page:

- ▶ Use [▼] / [▶] or [▲] / [◀] to select the required GUI element (e.g. number field, options field).
  - ▷ The selected screen element has focus (orange frame).
  - ▷ Selected element is enabled and can be changed.
- ▶ Enable or disable the selected screen element (e.g. option field) with [RETURN]
  - or -
  - Increment or decrement the value of the selected operating element using [▲] or [▼]
  - ▷ The changes are displayed
- ▶ Select  to save the changes.
- ▶ Change to the higher level menu page with .

### Operation: enter text with the on-screen keyboard

The user interface of the runtime system has an on-screen keyboard. It appears automatically when the user is supposed to enter characters.





The on-screen keyboard is a QWERTY keyboard. This setting cannot be changed.

When entering the password: To protect sensitive information, only the character entered last appears in clear text. All other characters are replaced by an \*.

To enter the text with the on-screen keyboard:


- ▶ Use [▲] / [▼] and [►] / [◄] to select the button of the required sign.

- ▷ The selected button has the focus (orange frame).


- ▶ Enable the selected button with [RETURN].

- ▷ The entered character appears in the text field.

- ▶ Repeat the process to enter all required characters.

- ▶ Confirm the password with 

- or -






Change to the previous setup screen with .

## Navigation keys

The navigation key element includes the following keys:

- 4 navigation keys ([▲], [▼], [◄], [►])
- 1 RETURN key (centre)

Certain navigation functions can be done both with the function keys and with the key element. The following table shows the functions of the navigation key element:

Navigation keys	Key	Function
	[▲]	Move up Select previous menu element
	[◄]	Move to the left Select previous menu element
	[►]	Move to the right select next menu element
	[▼]	Move down select next menu element
	[RETURN]	activate menu item / button

## touch screen


In the setup mode the touch screen functionality of the device is activated.

Certain navigation functions can be executed either via the function keys or via the touch screen. The user can activate the following operating elements by means of the touch screen:

- Buttons
- Menu items
- Keys of the on-screen keyboard
- Symbols in the file directory
- Input fields

## Set-up: Enter password

To enter the password for unrestricted access to the set-up menu:

- ▶ Enter the correct password. Operation: enter text with the on-screen keyboard (→ [24](#))
- ▶ Confirm the password with .
- ▷ The setup menu appears.
- ▷ The user has unrestricted access to all setup functions.



Full access to all setup function is only valid until the user exits the setup menu.

- ▶ When the setup menu is called up, enter the password again!

## 5.1.3 Connection

- ▶ Enable [Connection].
  - ▷ The [Connection] menu appears:
- ▶ Select one of the following sub-menus:

Designation	Meaning
[LAN & Ethernet]	Configuring the Ethernet interfaces (→ <a href="#">26</a> )

## Configuring the Ethernet interfaces



Standard settings of the IP address:

IP address = 192.168.82.247





Subnet mask = 255.255.255.0

Gateway address = 192.168.82.1

To configure the Ethernet interfaces:

### Select the menu page

- ▶ Navigation path: [Launch Setup] > [Connection] > [LAN & Ethernet]
- ▷ The menu screen shows the following information:

Designation	Meaning	Possible values
[Bridge mode]	Switch on/off bridge mode. If it is switched on, both Ethernet interfaces of the device are operated in bridge mode. They do not share an IP configuration.	 / 
[Ethernet 0] / [Ethernet 1]	Selection of the Ethernet interface	-
[DHCP mode]	Switch on/off DHCP mode	 / 
[IP Address]	IP address of the device's Ethernet interface	e.g. 192.68.82.247
[Netmask]	Subnet mask of the network segment	e.g. 255.255.255.0
[gateway]	IP address of the network gateway	e.g. 192.168.82.1


## Enable bridge mode

- ▶ Enable the [Bridge Mode] control field
- ▷ Bridge mode is enabled.

### Enable DHCP mode

- ▶ Enable the [DHCP Mode] control field
- ▷ DHCP mode is enabled.
- ▷ The device receives the IP address settings from a DHCP server in the network.

### Changing IP parameters



- ✓ The DHCP mode is inactive.
- ▶ Select the [Ethernet 0] / [Ethernet 1] interface (except in bridge mode).
- ▶ Highlight the [IP Address] input box.
  - ▷ Number pad appears.
- ▶ Enter IP address with the number pad.
- ▶ Proceed with the input boxes [Netmask] and [Gateway] as described.
- ▶ Save the settings with the  button.
- ▷ The IP address settings have been changed.

## 5.1.4 Backup: Creating a data backup


To store data from the device on a USB memory:

- ▶ connect USB memory to the USB interface of the device.
- ▶ Enable [Backup].
  - ▷ The [Backup] menu appears.
- ▶ Select the data and settings to be backed up:

Designation	Meaning
[Retain Memory]	Data that is stored on the FLASH memory of the device.
[Ethernet Settings]	Settings of the Ethernet interface.
[Brightness Setting]	Brightness settings.
[Screen Orientation Setting]	Display orientation settings (rotation).

- ▶ Adopt the value with .
  - ▷ The image [Select USB] appears.
- ▶ Select / highlight the USB memory and start the backup process with .
  - ▷ The data will be transferred to the connected USB memory.
  - ▷ The data is stored in a \*.swu file.
  - ▷ The backup progress is visualised.
  - ▷ When the backup is finished, a message appears to indicate whether the backup was successful.



The recovery mode can be used to restore the backup (\*.swu). Updating the runtime system (→  14)

## 5.1.5 System setup

- ▶ Enable [System Setup].
  - ▷ The [System Setup] menu appears:
- ▶ Select one of the following sub-menus:

Designation	Meaning
[Date and Time]	Date and time (→ □ 28)
[Password]	Setting the password (→ □ 29)
[Boot Screen]	Adjusting the boot screen (→ □ 31)
[Display Brightness]	Adjusting the display brightness (→ □ 32)
[Display Orientation]	Adjusting the display orientation (→ □ 32)
[Touch Configuration]	Configuring the touch screen (→ □ 32)

### Date and time

#### Setting the time zone

To set date and time (system time):

##### 1 Select the menu page

- ▶ Navigation path: [Launch Setup] > [System Setup] > [Date and Time] > [Time Zone] tab
  - ▷ Menu page shows the following information:

Designation	Meaning	Possible values
[Time Zone Category]	Pre-selection of the time zone region. The selection of a region filters the [Time Zone] list.	e.g. All, America, Asia, Australia, Europe, US
[Time Zone]	Time zone	e.g. US/Michigan, Europe/London


##### 2 Select the time zone region

- ▶ Select the [Time Zone Category] list.
  - ▷ The highlighted list has an orange frame.
- ▶ Use [▲] / [▼] to set the required value.
- ▶ Confirm with [RETURN].

##### 3 Change the time zone

- ▶ Select the [Time Zone] list.
- ▶ Use [▲] / [▼] to set the required value.
- ▶ Confirm with [RETURN].

##### 4 Store changed values

- ▶ Adopt and save the changed values with .

### Setting date and time

To set date and time (system time):

##### 1 Select the menu page

- ▶ Navigation path: [Launch Setup] > [System Setup] > [Date and Time] > [System Time] tab


▷ Menu page shows the following information:

Designation	Meaning	Possible values
[Year]	Year	2000 ... 2100
[Month]	Month	1 ... 12
[Day]	Day	1 ... 31
[Hour]	Hour	0 ... 23
[Min]	Minute	0 ... 59
[Sec]	Second	0 ... 59
[Time Zone]	Time zone	List

## 2 Changing date and time



- ▶ Select the required number field.
  - ▷ The highlighted number field has a black frame.
- ▶ Use [▲] / [▼] to set the required value.
- ▶ Optional: Repeat step 2 to change the values of additional groups of numbers.

## 3 Store changed values

- ▶ Adopt and save the changed values with .

## Setting the password

The menu page [Launch Setup] > [System Setup] > [Password] offers access to the following functions:

- Changing the system password (→  30)
- Enabling / disabling password protection (→  30)



On delivery, the password protection for the setup menu is disabled.

If the password protection is enabled after the first setup, the password set at the factory will be valid: `pdm3`

- ▶ To ensure effective protection, change this password set at the factory!



The programming software CODESYS can access the IEC project saved on the device even with activated password protection.

## Login data




The following login data is factory-preset:

User name/login: `root`

Password: `pdm3`

These settings apply e.g. to:

- Access to the device via FTP, Telnet, SSH
- Access to the setup mode, provided that password protection is activated (→  30).

---

## Changing the system password



If the password is forgotten, it is necessary to carry out a recovery update.

- ▶ Carry out a device update Updating the runtime system of the device (→ 13)
- ▶ restore stored data, if necessary Updating the runtime system (→ 14)



Notes on using the on-screen keyboard: Operation: enter text with the on-screen keyboard (→ 24)

To change the system password:

### 1 Select the menu page

- ▶ Navigation path: [Launch Setup] > [System Setup] > [Password]
- ▶ Continue with [Change Password] button.
  - ▷ The menu page shows the input box [Enter Old Password] and the on-screen keyboard.

### 2 Enter old system password

- ▶ Enter the old password character by character. Standard password: Login data (→ 29)
- ▶ Confirm the entry with the icon.



If the user enters the wrong password, the following error message appears: Incorrect password!

- ▶ Enter the password again!

### 3 Entering a new system password

- ▶ Enter the new password character by character.
- ▶ Confirm the entered password with .
- ▶ Repeat the new password and confirm it with .
- ▷ The info field shows the success message: Password is changed successfully.
- ▷ The new password is valid.



If the user enters two different new passwords, the following error message appears: Password doesn't match!

- ▶ Enter the password again!

## Enabling / disabling password protection



On delivery, the password protection for the setup menu is disabled.

If the password protection is enabled after the first setup, the password set at the factory will be valid: `pdm3`

- ▶ To ensure effective protection, change this password set at the factory!



The programming software CODESYS can access the IEC project saved on the device even with activated password protection.

To activate/deactivate the password protection of the device :

### 1 Select the menu page

- ▶ Navigation path: [Launch Setup] > [System Setup] > [Password]
  - ▷ The control field [Enable Password] shows the status of the password protection.

### 2 Setting the password protection

- ▶ Highlight the control field [Enable Password] and set it as required.
  - ▷ The last valid password is requested.
- ▶ Enter password
- ▷ Password protection will be enabled / disabled.



The password protection will be active the next time the setup menu is opened.

- ▶ Use to return to the previous menu screen until you have reached the setup main menu page.
- ▶ Quit the setup main menu with .
- ▶ Start the setup again with the [Launch Setup] button.

## Adjusting the boot screen



The image for the boot screen can also be loaded to the device via the image collection in the CODESYS project and enabled using a function block. Adjusting the boot screen (→ [53](#))

To set the image for the boot screen:

### 1 Select the menu page

- ▶ Navigation path: [Launch Setup] > [System Setup] > [Boot Screen]
  - ▷ Menu page shows the following information:

Designation	Meaning
[Recently set boot screen]	Currently set image for the boot screen

### 2 Changing the image

- ▶ Press .
  - ▷ The file browser on the device appears.
- ▶ Select the image file or change the file path with .
- ▶ Highlight the image file.
- ▶ Adopt the image file with .
- ▷ The image file for the boot screen has been changed.



Information on the image file for the splash screen:

- Image format: BMP 24 bit version 3.
- RLE compression and gzip compression are allowed.
- Recommended approach: Storage of the image file using MS Paint as 24 bit BMP.
- Recommended maximum image size: display resolution → Data sheet

- The file path is case-sensitive. Linux is case sensitive.
- The file name may only contain lower case letters.
- Transfer the new file to the unit using the CODESYS file browser or in setup mode.
- Maximum file size: unlimited. The image file uses a part of the available memory for the user application.

## Adjusting the display brightness

To adjust the display brightness:

### 1 Select the menu page


► Navigation path: [Launch Setup] > [System Setup] > [Display Brightness]

▷ Menu page shows the following information:

Designation	Meaning	Possible values	
[Display Brightness]	Relative brightness of the screen (value in %)	0 ... 100	Minimum brightness  Maximum brightness

### 2 Changing the display brightness

► Set the slider with the required value.

► Press  to verify the set value.

### 3 Saving the changed value

► Press  to save the set value.

## Adjusting the display orientation

To adjust the orientation of the display:

### 1 Select the menu page

► Navigation path: [Launch Setup] > [System Setup] > [Display Orientation]

▷ Menu page shows the following information:

Designation	Meaning
[Landscape Right]	Landscape format, keys on the right
[Landscape Left]	Landscape format, keys on the left
[Portrait]	Portrait format, bottom keys
[UpSide Down]	Portrait format, top keys

### 2 Changing the display orientation

► Adjust the required orientation.

▷ The device will reboot after an enquiry.

## Configuring the touch screen

To adjust the touch screen (if the device has one):

### 1 Select the menu page


► Navigation path: [Launch Setup] > [System Setup] > [Touch Configuration]

▷ The menu page shows the following setting options:







Designation	Meaning
[Water Optimized (default)]	Optimised for operation in wet conditions.
[Glove Optimized]	Optimised for use with gloves.

## 2 Changing the optimisation type of the display

- ▶ Set the required optimisation type.
- ▶ Press  to save the set value.



### 5.1.6 Device Diagnostic

- ▶ Enable [Device Diagnostic].
  - ▷ The menu [Device Diagnostic] appears:
- ▶ Select one of the following sub-menus:

Designation	Meaning
[CAN test]	CAN test (→  ) 33)
[Audio test]	Audio test (→  ) 35)
[Camera Test]	Camera Test test the camera image (→  ) 36)
[Keyboard Test]	Keyboard test: test the keyboard (→  ) 36)



### CAN test

- ▶ Navigation path: [Launch Setup] > [Device Diagnostic] > [CAN Test]
  - ▷ The menu page [CAN Test] appears.
  - ▷ Menu page shows the following information:

Designation	Meaning	Possible values
[Bus Number]	Enter the CAN bus interface.	1 ... 4
[Baud rate]	Set the baud rate of the CAN bus.	20 kbit/s 33.3 kbit/s 50 kbit/s 83.3 kbit/s 100 kbit/s 125 kbit/s 250 kbit/s 500 kbit/s 800 kbit/s 1000 kbit/s
[Tx Frame Counter]	Number of messages transmitted	
[Rx Frame Counter]	Number of messages received	
[CAN Tx-ID (hex)]	CAN ID of the sender (as hexadecimal number)	00000000 ... FFFFFFFF
[Extended Frame:]	Enable / disable the extended frame format	 / 
[Tx Bytes (hex)]	CAN message to be sent (8 bytes) The content of each byte can be set separately (as a hexadecimal number).	per byte: 00 - 0 ... FF=255
[Receive]	Table view of the received CAN messages: [ID], [Frame], [Type], [DLC], [Bytes]	

---

The menu page [Launch Setup] > [Device Diagnostic] > [CAN Test] offers access to the following functions:

- CAN test: transmitting data (→  34)
- CAN test: Data received (→  34)

### **CAN test: transmitting data**

To test the data transmission via a CAN interface:

#### **1 Select the menu page**

- ▶ Navigation path: [Launch Setup] > [Device Diagnostic] > [CAN Test]
  - ▷ The menu page [CAN Test] appears.




#### **2 Select CAN bus**

- ▶ Enter the number of the CAN bus in the input box [Bus Number].
- ▶ Set the baud rate in the [Baudrate] list.

#### **3 Set the test message and the transmitter ID**

- ▶ Set the CAN ID of the transmitter in [CAN Tx-Id (hex)].
- ▶ Enable / disable extended frame format in the control field [Extended Frame].
- ▶ Set the test CAN message to be sent, segment by segment in [Tx Bytes (hex)].
  - ▷ The test message and the transmitter ID are set.

#### **4 Sending a test CAN message**

- ▶ Send the test message with the  symbol.
- ▶ The device tries to send the CAN message via the selected CAN bus.
  - ▷ The number of the CAN messages sent since the beginning of the transfer appears in [Tx Frame Counter].
  - ▷ If successful:
    - ▷ A success message appears.
- ▶ Stop cyclical transmission of the CAN message with .
- ▷ If not successful:
- ▶ Stop cyclical transmission of the CAN message with .
- ▶ Check the connection to the CAN bus.
- ▶ Repeat the operation.

### **CAN test: Data received**

To test the data reception via a CAN interface:




#### **1 Select the menu page**

- ▶ Navigation path: [Launch Setup] > [Device Diagnostic] > [CAN Test]
  - ▷ The menu page [CAN Test] appears.

## 2 Select CAN bus

- ▶ Enter the number of the CAN bus in the input box [Bus Number].
- ▶ Set the baud rate in the [Baudrate] list.

## 3 Receiving the CAN message

- ▶ Start the reception of CAN messages on the selected CAN bus with .
  - ▷ The received CAN messages appear one after the other in hexadecimal format in the [Receive] table.
  - ▷ The number of the CAN messages received since the beginning of the transfer appears in [Rx Frame Counter].
  - ▷ If successful:
  - ▷ [Rx Frame Counter] shows a value > 0.
- ▶ Stop reception of CAN messages with .
  - ▷ If not successful:
  - ▷ [Rx Frame Counter] shows the value 0.
- ▶ Stop reception of CAN messages with .
- ▶ Check the connection to the CAN bus.
- ▶ Repeat the operation.

## Audio test



- ▶ For this function, connect an external loudspeaker to the unit. → Installation instructions

## 1 Select the menu page

- ▶ Navigation path: [Launch Setup] > [Device Diagnostic] > [Audio Test]
  - ▷ The menu page [Audio Test] appears.
  - ▷ Menu page shows the following information:

Designation	Meaning	Possible values
[Volume]	Set the volume	0 ... 100 %
[Balance]	Set the balance	Left / Middle / Right

## Adjusting and testing the sound volume and the balance



- ▶ For this function, connect an external loudspeaker to the unit. → Installation instructions

To set the balance and the sound volume:

## 1 Select the menu page

- ▶ Navigation path: [Launch Setup] > [Device Diagnostic] > [Audio Test]
  - ▷ The menu page [Audio Test] appears.

## 2 Setting the sound volume

- ▶ Enter the volume in the input box [Volume] or set it using the slider.

---

### 3 Setting the balance

- ▶ Set the balance in the option field [Balance].

### 4 Testing the sound volume and the balance

- ▶ Test the volume and the balance with .

### Camera Test test the camera image

- ▶ Navigation path: [Launch Setup] > [Device Diagnostic] > [Camera Test]
  - ▷ The menu page [Camera Test] appears.
  - ▷ The menu page shows the images of the connected cameras 1 ... 4.

### Keyboard test: test the keyboard

- ▶ Navigation path: [Launch Setup] > [Device Diagnostic] > [Keyboard Test]
  - ▷ The menu page [Keyboard Test] appears.
- ▶ Press the required key of the integrated key pad.
  - ▷ The key that is pressed will be displayed.

### 5.1.7 Device Info: show device information


To display the device information:

- ▶ Navigation path: [Launch Setup] > [Device Info]
  - ▷ Menu page shows the following information:

Name	Description
[Firmware]	Firmware version
[Serial number]	Serial number of the device
[MAC Address]	MAC addresses of the Ethernet interfaces
[External Voltage (supply clamp 15)]	Value of the connected supply voltage on terminal 15 in mV
[Internal Voltages]	Internal voltage values
[Board Temperature]	Temperature on the PCB
[CPU Core Temperature]	CPU core temperature
[Memory Usage Harddisk]	Memory size / memory used

### 5.1.8 Removing USB: Removing the USB stick safely

To remove a USB stick safely:

- ▶ Navigation path: [Launch Setup] > [Remove USB]
  - ▷ USB sticks connected to the device are shown.
- ▶ Check out the USB stick with the  icon.
  - ▷ The use stick can now be removed safely.

---

## 5.2 Starting the PLC application

- ▶ Starting the set-up mode: Starting the set-up mode (→ [22](#))
- ▶ Highlight the [Load Application] button.
- ▶ Start the application with [RETURN].
  - ▷ If a valid application program is saved: The application starts.
  - ▷ If no valid application program is saved: A blank screen appears.
- ▶ Transfer a valid application program to the device using CODESYS.
- ▶ Reboot the device (supply voltage OFF > ON).

## 5.3 Quitting the setup, rebooting the device

Rebooting the device (→ [23](#))

---

## 6 System configuration

### 6.1 Configuring the PLC



The configuration of the PLC of the ecomatDisplay is made via the "Generic device editor" of the CODESYS programming system.

- ▶ Familiarise yourself with the following CODESYS functions.  
Device editor: → Online help > CODESYS Development System > Reference user interface > Objects > Object 'Device' and Generic Device Editor

The configuration of the PLC is made via the node [Device (ecomatDisplay)] of the device tree.

To configure the PLC of the ecomatDisplay:

- ▶ In the device tree: Double-click [Device (ecomatDisplay)]
- ▷ The editor window shows device editor of the PLC of the ecomatDisplay.

### 6.2 Adding a font

Load a font (True Type Font) to the controller:

- ▶ connect the device via CODESYS or winSCP. Accessing the Linux system of the device (→ 21)
- ▶ Copy the file with the True Type Font ( \*.ttf ) to the following folder on the device:  
`/usr/share/fonts/`
- ▶ Restart the device.
- ▷ All fonts in the folder will be installed and can be used.



If the font is also reinstalled in Windows:

- ▶ If CODESYS is open, restart it so that the new font is available in CODESYS.

### 6.3 Configuring CAN interfaces



- ▶ Familiarise yourself with the following CODESYS functions:  
CAN-based fieldbuses → Online help > Fieldbus support > CAN-based fieldbuses

The device has CAN interfaces.

Each CAN interface supports the following protocols:

- RawCAN (CAN Layer 2)
- CANopen Manager
- CANopen device
- J1939 Manager



- ▶ Observe the notes about task configuration! Configuring task processing (→ 69)
- ▶ Add an individual CANbus device to the device tree for each CAN interface used in the application!

### 6.3.1 Device description files (EDS files)



If required:

- ▶ Download the EDS files with the device descriptions for ifm devices from the ifm website:  
[www.ifm.com](http://www.ifm.com)
- ▶ Install the EDS files via the device repository in CODESYS.

### 6.3.2 Adding and configuring a CANbus



- ▶ Familiarise yourself with the following CODESYS functions:  
CANbus settings → Online help > Fieldbus Support > CAN-Based Fieldbuses > Tab "CANbus - General"
- CANbus-I/O image: → Online help > Fieldbus Support > CAN-Based Fieldbuses > Tab "IO-Mapping"
- ▶ Add an individual CANbus device to the project tree for each CAN interface.

#### Add CANbus:

- ▶ In the device tree: Right-click on [CAN].
  - ▷ Context menu appears.
- ▶ Select [Add Device ...].
  - ▷ The window [Add Device] appears.
- ▶ Set the following values:  
[Vendor]: Select ifm electronic.  
In the table: Select [ifmCANbus].  
[Name]: Enter unambiguous name for the CAN interface.
- ▶ Click on [Add Device] to add the selected device to the project.
- ▶ Click on [Close] to close the window.
- ▷ CODESYS adds CAN interface to the device tree.

#### Configure CANbus:

- ▶ In the device tree: Double-click on the added CAN node.
  - ▷ Editor window shows setting options of the CAN interface.
- ▶ Select the CAN ID of the CAN network in the field [Network].
- ▶ Select the transmission rate of the CAN network in the list [Baudrate (bit/s)].
- ▶ Continue with:
  - RawCAN: configuring CANLayer 2 (→ [39](#))
  - CANopen: configuring CANopen Manager (master) (→ [40](#))
  - CANopen: configuring the CANopen Device (slave) (→ [40](#))
  - J1939: configuring J1939 Manager (→ [41](#))

### 6.3.3 RawCAN: configuring CANLayer 2

No further actions are required to use a CAN interface as CANLayer 2 (RawCAN). The user can access the CAN interface directly in the application.



- ▶ More information about programming of the RawCAN interface: Using RawCAN (CAN Layer 2) (→ [64](#))

### 6.3.4 CANopen: configuring CANopen Manager (master)



- ▶ Familiarise yourself with the following CODESYS functions:  
CANopen Manager: → Online Help > Fieldbus Support > CAN-Based Fieldbuses > CANopen > CANopen Manager

The configuration of the CANopen Manager is based on the CANopen Stack of the CODESYS 3.5 programming system.

- ✓ Add and configure CAN interface: Adding and configuring a CANbus (→ [39](#))
- ▶ In the device tree: Right-click on the added [CAN] node
  - ▷ Context menu appears.
- ▶ Select [Add Device...] in the context menu.
  - ▷ Dialogue window [Add Device] appears.
- ▶ Set the following values:  
[Vendor]: [<All vendors>] In table: Select [CiA CANopen] > [CiA CANopen Manager] > [CANopen Manager].  
[Name]: Enter unambiguous name.
- ▶ Click on [Add Device] to add the selected device to the project.
- ▶ Click on [Close] to close the window.
- ▷ CODESYS adds CANopen Manager to the device tree.
- ▶ In the device tree: Double-click on the added CANopen Manager
  - ▷ Editor window shows configuration options.
- ▶ Configure CANopen Manager as requested.
- ▶ Save the project to adopt all changes.



- ▶ More information about programming of the CANopen interface: Using CANopen (→ [64](#))

### 6.3.5 CANopen: configuring the CANopen Device (slave)



- ▶ Familiarise yourself with the following CODESYS functions:  
CANopen Local Device: → Online Help > Fieldbus Support > CAN-Based Fieldbuses > CANOPEN > CANopen Local Device

The configuration of the CANopen Device is based on the CANopen Stack of the CODESYS 3.5 programming system.

- ✓ Add and configure CAN interface: Adding and configuring a CANbus (→ [39](#))
- ▶ In the device tree: Right-click on the added [CAN] node
  - ▷ Context menu appears.
- ▶ In the context menu: Select [Add Device...].
  - ▷ Dialogue window [Add Device] appears.



- ▶ Set the following values:  
[Vendor]: [<All vendors>]  
In table: Select [CiA CANopen] > [CiA Local Device] > [CANopen Device].  
Name: Enter unambiguous name.

▶ Click on [Add Device] to add the selected device to the project.

▶ Click on [Close] to close the window.

▷ CODESYS adds CANopen Device to the device tree.

▶ In the device tree: Double-click on the added CANopen Device

▷ Editor window shows configuration options.

▶ Configure CANopen Device as requested.

▶ Save the project to adopt all changes.



More information about programming of the CANopen interface: Using CANopen (→ [64](#))

### 6.3.6 J1939: configuring J1939 Manager



- ▶ Familiarise yourself with the following CODESYS functions:  
J1939 manager → Online Help > Fieldbus Support > CAN-Based Fieldbuses > J1939 > J1939 Manager

The configuration of the J1939 Manager is based on the J1939 Stack of the CODESYS 3.5 programming system.

To configure a CAN interface as J1939 Manager:

✓ Add and configure CAN interface. Adding and configuring a CANbus (→ [39](#))

▶ In the device tree: Right-click on the added [CAN] node

▷ Context menu appears.

▶ Select [Add Device...] in the context menu.

▶ Dialogue window [Add Device] appears.

▶ Set the following values:

[Vendor]: <All vendors>

In table: Select [SAE J1939] > [SAE J1939 Manager].

[Name]: Enter unambiguous name.

▶ Click on [Add Device] to add the selected device to the project.

▶ Click on [Close] to close the window.

▷ CODESYS adds J1939 Manager to the device tree.

▶ In the device tree: Double-click on the added J1939 Manager

▷ Editor window shows configuration options.

▶ Configure J1939 Manager as requested.

▶ Save the project to adopt all changes.

## 7 Programming

### 7.1 Objects of the PLC application with ecomatDisplay template

All objects of a PLC application are listed as subelements of the node [Application] in the device tree. In the basic configuration of the template Creating a CODESYS project (→ 18), a PLC application includes the following objects:

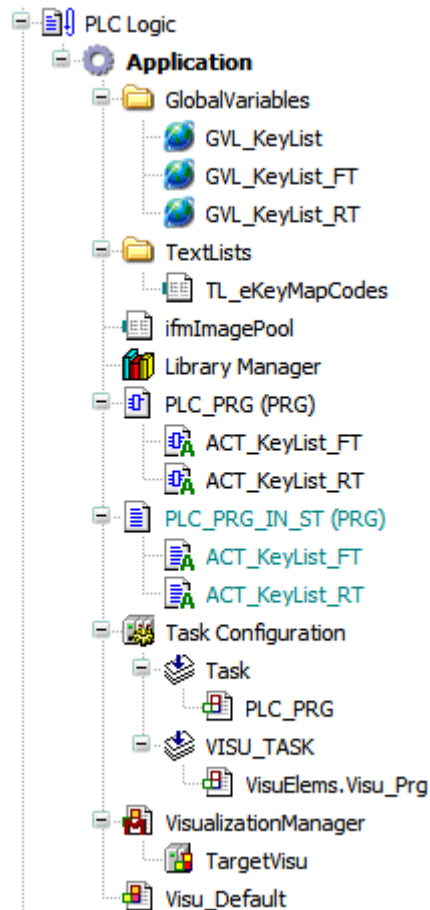


Fig. 2:

Legend:

Object	Description
Application	Container for objects of a PLC application.
GlobalVariables	Folder with global lists of variables.
GVL_KeyList	Global list of variables for the evaluation of the device keys.
GVL_KeyList_FT	Global list of variables for the evaluation of the device lists as a falling edge.
GVL_KeyList_RT	Global list of variables for the evaluation of the device keys as rising edge.
TextLists	Folder for text lists.
TL_eKeyMapCodes	Text list with mapping codes of the keys.
ifmImagePool	Image collection for the visualisation.
Library manager	Provides access to standard and device-specific function libraries:
PLC_PRG(PRG)	Offers access to the PLC application in the programming language Function Block Diagram (FBD).
PLC_PRG_IN_ST(PRG)	Offers access to the PLC application in the programming language Structured Text (ST). By default, excluded by the build.
Task configuration	Provides access to the settings of the task processing:

Object	Description
Task	Main task with assigned PLC_PRG.
VISU_TASK	Visualisation task with assigned visualisation.
Visualisation manager	Offers access to the properties of the visualisation.
TargetVisu	Offers access to the properties of the display.
Visu_Default	Provides access to the visualisation page.

## 7.2 Creating a PLC application



- Familiarise yourself with the following CODESYS functions:  
→ Online Help > CODESYS Development System > Programming of 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)]
  - ▷ The editor window shows the input mask of the selected programming language.
- Enter program code.

### 7.2.1 Supported programming languages

The following programming languages according to IEC 61131 are supported by the ifm function libraries:

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

### 7.2.2 PLC\_PRG in FUP and ST

The project template contains the PRGs PLC\_PRG (PRG) and PLC\_PRG\_IN\_ST (PRG). Objects of the PLC application with ecomatDisplay template (→ 42)

The PRG PLC\_PRG (PRG) is programmed in Function Block Diagram (FBD) and considered by default during the compiling process (shown in black characters).

The PRG PLC\_PRG\_IN\_ST (PRG) is programmed in Structured Text (ST) and by default not considered during the compilation process (shown in green characters).

Both PRGs have the same functionality.

If required, the PRG PLC\_PRG (PRG) can be disabled and the PRG PLC\_PRG\_IN\_ST (PRG) can be enabled for the compilation process.

Disable the PRG PLC\_PRG (PRG) :

- Enable with right-click on [PLC\_PRG (PRG)] > [Properties] > [Build] > [Exclude from build].
  - ▷ The PRG [PLC\_PRG (PRG)] is green in the device tree and blocked for compiling.
- Rename [PLC\_PRG (PRG)] to [PLC\_PRG\_IN\_FUP (PRG)].

- ▷ The PRG programmed in FBP is now disabled

Enable the PRG `PLC_PRG_IN_ST` (PRG) :

- ▶ Disable with right-click on [PLC\_PRG\_IN\_ST (PRG)] > [Properties] > [Build] > [Exclude from build]
  - ▷ The PRG [PLC\_PRG (PRG)] is black in the device tree and enabled for compiling.
- ▶ Rename [PLC\_PRG\_IN\_ST (PRG)] to [PLC\_PRG (PRG)].
- ▷ The PRG programmed in ST is now enabled.

### 7.2.3 Available memory

Memory area	size
flash memory	2...8 GB, depending on the device type → Data sheet
RAM memory	1 GB

### 7.2.4 Supported variable types



- ▶ Familiarise yourself with the following CODESYS functions:  
Local variables → Online Help > CODESYS Development System > Reference Programming > Variables > Local Variables – VAR

Global variable list → Online Help > CODESYS Development System > Reference Programming > Variables > Local Variables > VAR\_GLOBAL

Network variables: → Online Help > CODESYS Development System > Working with Controller Networks > Network Variables

The device supports the following variable types:

Variable type	Declaration	Scope of validity	Memory behaviour
local	In the declaration part of the POU	Applies only to the POU in which it has been declared	volatile
global	In the global variable list (GVL)	Applies to all POUs of the project	volatile
global retain			non volatile
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.	volatile



CAN network variables are not supported!

### 7.2.5 Persistent variables

Values of persistent variables (retain variables or remanent variables) are retained in memory after the unit is switched off and restarted. Size of the retentive memory → Technical data

Define persistent variables only in the `Persistent Variable List` below the [Application] :

- ▶ Right-click on [Application] > [Add Object] > [Persistent Variables...].
  - ▷ The [Add Persistent Variables] window appears.
- ▶ Enter [Name].

► Click on [Add].

▷ The new **Persistent Variable List** will be created below the [Application].

Create persistent variables in the **Persistent Variable List** as follows:

► Open the **persistent variable list** in the editor by double-clicking.

► Enter and save the definition of the persistent variables.

Example:

```
VAR_GLOBAL_PERSISTENT RETAIN
```

```
retain_var1: INT;
```

```
retain_var2: REAL;
```

```
retain_var3: STRING;
```

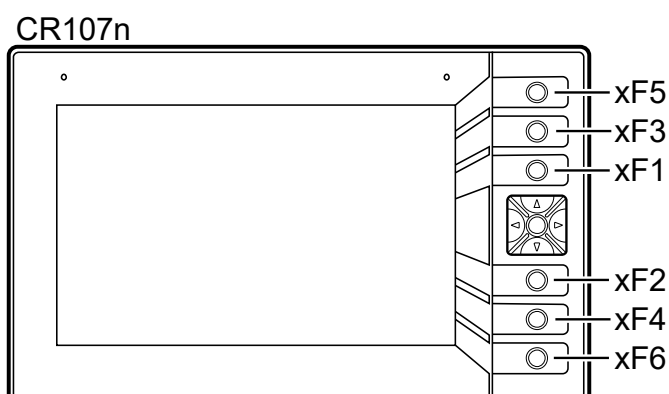
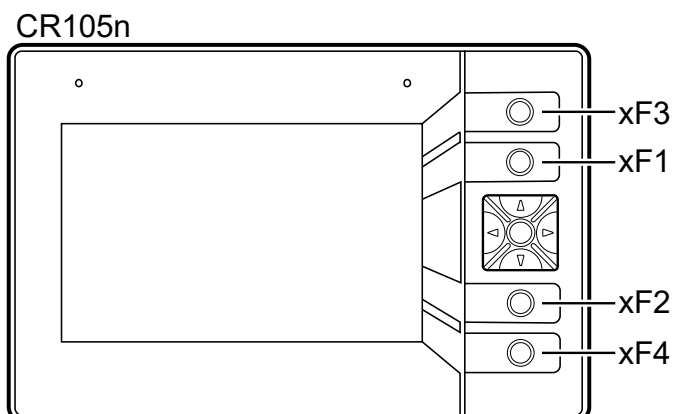
```
END_VAR
```

## 7.2.6 Symbol names of the operating elements

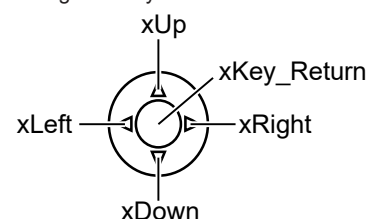
Certain symbol names are assigned to the operating elements of the device. By means of these symbol names the programmer can define certain actions and functions to be triggered upon actuation of the correspondent operating element. Defining functions for operating elements (→ 54)

The following figure shows the symbol names of the individual operating elements:

Function keys:



Navigation keys



## 7.2.7 Procedure

In principle, there are two options to create a project for display devices:

Order	Advantages	Disadvantages
<b>Visualisation first</b> , then the PLC application	<ul style="list-style-type: none"><li>• In the program it is possible to cross-reference to the finished images.</li><li>• When testing the PLC application, the images already exist.</li></ul>	The PLC parameters and variables required in the visualisations have not yet been defined.
<b>PLC application first</b> , then the visualisation	All parameters and variables are defined in the PLC program before they are referred to in the visualisations.	<ul style="list-style-type: none"><li>• The parameters from the images (image number, key, LED, etc.) must be found elsewhere.</li><li>• The PLC application can only be tested after the visualisation has been created.</li></ul>



► **Before** starting, design a structure as precisely as possible for the visualisation and its contents!

## 7.3 Using ifm function libraries

ifm 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
ifmDevice	Device-specific functions, data structures, enumeration types and global variables
ifmFileUtil	File and help functions
ifmRawCAN	Functions for use of the CAN interfaces as CAN Layer 2



► Detailed information about the ifm function libraries: ifm function libraries (→ [75](#))

### 7.3.1 Configuring the device

The following function elements are available to configure the device:

Name	Description	Reference
HideSplashScreen	Show/hide splash screen.	Common (→ <a href="#">94</a> )
LoadSplashScreen	Load individual splashscreen.	
SetSupplySwitchMode	Set behaviour for auto shutdown of the device.	
GetDeviceOrientation	Determine the set device orientation.	
SetDeviceOrientation	Set device orientation.	
GetSysInfo	Read system information.	System Information (→ <a href="#">173</a> )

### 7.3.2 Controlling the device

The following function elements are available to control the device:

Name	Description	Reference
ShutdownDevice	Shut down / reboot the device.	Common (→ <a href="#">94</a> )
BootIntoRecovery	Call up recovery mode.	
SetStatusLEDstate	Controlling the status LED of the device.	Status LED (→ <a href="#">159</a> )

Name	Description	Reference
LinuxSysCallAsync	Transmit command to the Linux operating system of the device. Asynchronous execution of the command.	System Commands (→ <a href="#">165</a> )
LinuxSysCallAsync2	Transmit command to the Linux operating system of the device. Asynchronous execution of the command.	
LinuxSysCallSync	Transmit command to the Linux operating system of the device. Synchronous execution of the command.	
LinuxSysCallSync	Transmit command to the Linux operating system of the device. Synchronous execution of the command.	

### 7.3.3 Executing and configuring audio functions

To execute and to configure audio functions of the device, the following function elements are available:

Name	Description	Reference
AudioPlayer	Playing audio files.	Audio (→ <a href="#">86</a> )
AudioRecorder	Recording audio files.	
GetAudioSettings	Reading the audio settings.	
SetAudioSettings	Configuring the audio settings.	

### 7.3.4 Configuring the Ethernet interface

The following function elements are available to manage the settings of the Ethernet interface of the device:

Name	Description	Reference
GetEthernetInterfaces	Provide a list with the available Ethernet interfaces.	Ethernet (→ <a href="#">108</a> )
GetIPSettings	Read IP settings of the Ethernet interface	
SetIPSettings	Change IP settings of the Ethernet interface	
GetEthernetBridgeConfig	Read Ethernet bridge mode settings	
SetEthernetBridgeConfig	Change Ethernet bridge mode settings	



- The current IP settings of the Ethernet interface can be read in the online mode via the device tree Displaying system information (→ [74](#))

### 7.3.5 Configuring device keys

The following function elements are available to configure the integrated device keys:

Name	Description	Reference
ControlAllKeyLEDs	Set all key LEDs.	Keypads (→ <a href="#">118</a> )
GetKeyMapping	Read mapping settings of the keypad.	
ResetAllKeyLED	Switch off all key LEDs.	
ResetKeyMapping	Reset mapping settings of the keypad.	
SetKeyAutoRepeat	Set auto repeat configuration of the keypad.	

Name	Description	Reference
SetKeyLED	Set the colour and the night mode of a key LED.	Keypads (→ <a href="#">118</a> )
SetKeyMapping	Set mapping configuration of the keypad.	

### 7.3.6 Configuring the device display

The following function elements are available to configure the device display:

Name	Description	Reference
LCDcontrol	Control display and background illumination.	LCD (→ <a href="#">132</a> )
GetBootupBacklight	Read the brightness settings for the boot process.	
GetLCD_Backlight	Read the brightness settings for normal operation.	
SetBootupBacklight	Adjust the brightness settings for the boot process.	
SetLCD_Backlight	Adjust the brightness settings for normal operation.	

### 7.3.7 Accessing device sensors and inputs/outputs



► Technical details about the sensors, inputs and outputs: → Installation instructions, data sheet.

To read the sensors and the inputs of the device and to write the outputs, the following function elements are available:

Name	Description	Reference
GetLightSensor	Read integrated light sensor.	Local IO (→ <a href="#">142</a> )
GetLightSensorCyclic	Read integrated light sensor cyclically.	
GetLocalInputs	Read local digital inputs.	
GetLocalInputsCyclic	Read local digital inputs cyclically.	
GetTemperatures	Read device temperatures.	
GetTemperaturesCyclic	Read device temperatures cyclically.	
GetVoltages	Read device voltages	
GetVoltagesCyclic	Read device voltages cyclically.	
SetLocalOutputs	Write local digital outputs.	

### 7.3.8 Configuring/reading system time

To configure and read the system time of the device, the following function elements are available:

Name	Description	Reference
GetAvailableTimeZones	Show a list of the available time zones.	System Time (→ <a href="#">179</a> )
GetSysTime	Read date, time and time zone of the device.	
GetSysTimeCyclic	Read date, time and time zone of the device cyclically.	
SetSysTime	Set the date and time of the device.	
SetTimeZone	Set the time zone of the device.	
GetNTP_Settings	Read the NTP server settings.	
SetNTP_Settings	Configure the NTP server settings.	



### 7.3.9 File management

The following function elements are available to manage (copy, delete) the files and directories:

Name	Description	Reference
USBstorageHandler	Manage USB device.	Storage (→ <a href="#">161</a> )
USBstorageHandlerMulti	Manage several USB devices.	
Copy_Device_To_USB	Copy files from the device to a USB memory device	ifmFileUtil.library (→ <a href="#">220</a> )
Copy_USB_To_Device	Copy files from a USB memory device to the device	
DeleteFile	Delete a file from the device	
FileCopy	Copy files on the device	
SyncFileMemory	Synchronise the contents of the FLASH memory and the RAM	

### 7.3.10 Configuring the touch screen

The following function elements are available to configure the touch screen:

Name	Description	Reference
DisableTouchScreen	Enable / disable touch screen functionality.	Touch (→ <a href="#">175</a> )
SetTouchOptimisationMode	Set optimisation mode for the touch screen.	

### 7.3.11 Setting and controlling the analogue camera

The following function element is available to control and configure the analogue camera:

Name	Description	Reference
AnalogueCameraWindowControl	Setting and controlling the analogue camera window.	Window Control (→ <a href="#">193</a> )

### 7.3.12 Setting and controlling the Ethernet camera

To control and configure an Ethernet camera (IP camera), the following function element is available:

Name	Description	Reference
IPCameraWindowControl	Set and control Ethernet camera windows.	IPCameraWindowControl (→ <a href="#">195</a> )

### 7.3.13 Configuring the PDF viewer

To control and configure a PDF viewer, the following function element is available:

Name	Description	Reference
PDF_Viewer	Set and control PDF viewer.	PDF_Viewer (→ <a href="#">198</a> )

### 7.3.14 Controlling image fields / making a screenshot

The following function elements are available to control and configure the field:

Name	Description	Reference
WindowControlBase	Control the field.	Window Control (→ <a href="#">193</a> )
Printscreen	Make a screenshot of the display content.	

### 7.3.15 Managing CSV files

The following function elements are available to manage (create, change, read) CSV files:

Name	Description	Reference
ReadCSV8Byte	Read the contents of a CSV file	Log File (→ <a href="#">234</a> )
WriteCSV8Byte	Write the contents of a CSV file	
WriteCSV8ByteHeader	Write header section of a CSV file	
ReadCSVData	Read the contents of a CSV file	
WriteCSVData_Linear	Writing the contents of a CSV file, linear mode	
WriteCSVData_Ring	Writing the contents of a CSV file, ring mode	
WriteCSVHeader	Write header section of a CSV file	

### 7.3.16 Using help functions

The user can use the following help functions:

Name	Description	Reference
GetMemoryInfoAsync	Display memory used of the device	Support (→ <a href="#">251</a> ) Function (→ <a href="#">257</a> ) Parameter File (→ <a href="#">230</a> )
ByteArray_To_String	Convert array from bytes into a character string	
Buffer_To_String	Convert array from bytes into an array of character strings	
Pack2Byte_To_Word	Convert 2 bytes into a word	
Pack4Byte_To_DW	Convert 4 bytes into a double word	
Word_To_2Byte	Convert word into 2 bytes	
_8Byte_To_CSV	Convert array from 8 bytes into CSV format.	
DW_To_4Byte	Convert DWORD into 4 bytes	
RTC_To_String	Provide operating time of the device as STRING	
GetFileSizeAsync	Display size of a file	
ReadParmSingleAsync	Read individual parameter set from a text file	
WriteParmSingleAsync	Write individual parameter set into a text file	
ANY_TYPE_TO_STRING	The function converts a data value into a string.	
ifmCONCAT	The function connects 2 strings and outputs the result in a string. (String length defined in ifmGCL.uiGenericLogSizeMax)	
ifmFind	The function returns the position of the string sStr2 in the string sStr1. (String length defined in ifmGCL.uiGenericLogSizeMax)	
ifmMID	The function reads a substring of another string. (String length defined in ifmGCL.uiGenericLogSizeMax)	

## 7.4 Using visualisations



- Familiarise yourself with the following CODESYS functions:  
→ Online help > CODESYS Visualization

The device supports the following visualisation types:

- Target visualisation
- Web visualisation

### 7.4.1 Settings in the project template

If you use the project template when creating the project Creating a CODESYS project (→ 18), the following elements for visualisation are already included in the device tree under [Application]:

Object	Description
VISU_TASK	Visualisation task
VisualisationManager	Object to manage the basic settings.
TargetVisu	Object to configure the target visualisation shown on the display.
WebVisu	Object to configure the web-based visualisation that is displayed in a web browser via network access.
Visu_Default	Object that contains a visualisation image.

These objects are pre-configured and could be adjusted.

### 7.4.2 Integrating external files



Possible external files: Audio files, PDF files, image files

External files can be integrated into the CODESYS project and loaded to the device:

- Right-click on [Application > Add Object > External File...]  
▷ The window [Add External File] appears.
- Select [File path]. Select more settings.
- Click on [Add].  
▷ The file will be added as an external file to the project.
- ▷ When loading the project to the device, CODESYS transfers external files to the following path to the device: /home/cds-apps/PlcLogic/Application/

#### File name conventions



- Note that the file path and file names are case-sensitive.
- Do not use spaces in file names and paths.
- Do not use special characters (e.g. |, \, :, (, ), &, ;, ., ,) in file names and paths.
- The maximum length of a file name is 255 characters.
- The file name must be unique in the destination folder.

### 7.4.3 Texts and fonts

- The smallest font size that can be easily read on the device is 8 point.

- Standard fonts are available on the device.
- By default, the following character sets/fonts are available on the machine: Latin script, Arabic script, Cyrillic script



- ▶ If required: Install more True Type Font fonts, e.g. for Asian languages.
- ▶ Install all font files required for visualisation on the PC and on the device. Adding a font (→ [38](#))

- Already translated texts can be copied from the source document to the text property of an object in the visualisation editor by drag&drop. The characters of installed languages (e.g. Cyrillic, Arabic) are retained.

#### 7.4.4 Language selection



- ▶ Familiarise yourself with the following CODESYS functions:  
Managing text in a text list:  
→ Online help > CODESYS Development System > Programming an application > Managing text in a text list

Implementing a language switch in the visualisation:

- ▶ Create a global text list with the texts used in the visualisation in the required languages. `DynamischeTexte`
- ▶ Connect the [Dynamic texts] property of the visualisation objects with the text list and the corresponding text ID.
- ▶ Create an option to switch languages, e.g. with a button.

#### Example of language switching by mouse click

Configuration of language switching with a button in the visualisation.

How to proceed:

- ▶ Drag a [button] from the tool window onto the visualisation.
- ▶ Click on [Configure...] in the [Properties] of the button under [Input configuration > OnMouseClicked].
- ▶ Highlight [Change Language] and adopt by clicking on [>].
- ▶ Highlight [Change Language] in the right-hand section.
- ▶ Click on [...] to open the window to assign the target language.
- ▶ Select the target language, e.g. "en", and click [OK].
- ▶ Complete the [Input configuration] and click [OK.]
- ▷ Language switching is configured for the button.
- ▷ The button triggers language switching in the visualisation if the corresponding elements are linked to a translated text list ([Properties > Dynamic texts])

#### Example Variable `CurrentLanguage`

The currently set language of a visualisation is in the variable `VisuElems.CURRENTLANGUAGE`, e.g. German "de", English "en", Chinese "zh-CHS".

The button triggers language switching of the visualisation if the corresponding elements are linked to a translated text list ([Properties > Dynamic texts]).

✓ `VisuElems.CURRENTLANGUAGE = de`

- ▶ Change value, e.g. in PRG: `VisuElems.CURRENTLANGUAGE := 'en';`

- ▷ The language of the visualisation switches.

### 7.4.5 Using image pools



- Familiarise yourself with the following CODESYS functions:  
Image collection: → Online Help > CODESYS Development System > Programming of Applications > Using Image Pools

To use own images in visualisations the user must first add the corresponding image files to an image collection in the application.

To add an image collection to a project:

- In the device tree: Highlight the node [Application].
- Select [Project] > [Add Object...] > [Image Pool...].
  - ▷ The window [Add Image Pool] appears.
- Enter the name of the image collection and confirm with [Add].
- ▷ Image collection appears in the project tree as a subelement of the application.

#### Image properties



The following image file formats are supported: BMP, JPG, PNG, SVG, TIF

#### Adjusting the boot screen

The image for the boot screen can be loaded to the device via the image collection in the CODESYS project and activated with the FB LoadSplashScreen (→ 96)

#### Adding an image for the boot screen to the image collection

- Double-click on the image collection.
  - ▷ Detailed view appears.
- Right-click on the empty row in the image collection > [Insert Image]
- Set the path to the image file.
- ▷ The image has been added to the image pool.

#### Load the image to the device

The image is loaded to the device together with the CODESYS project.

The image is loaded to the device in the following path: `/home/cds-apps/PlcLogic/visu/`

#### Setting an image with FB

- Paste FB `LoadSplashScreen` in PRG.
- Set the FB parameters and integrate the FB into the program logic.
- Enter the path and the file name of the boot screen image file at the function block input `sPathToSplash`, e.g. `'/home/cds-apps/PlcLogic/visu/testbild.bmp'` (incl. quotation marks, upper/lower case)



Information on the image file for the splash screen:

- Image format: BMP 24 bit version 3.
- RLE compression and gzip compression are allowed.
- Recommended approach: Storage of the image file using MS Paint as 24 bit BMP.

- Recommended maximum image size: display resolution → Data sheet
- The file path is case-sensitive. Linux is case sensitive.
- The file name may only contain lower case letters.
- Transfer the new file to the unit using the CODESYS file browser or in setup mode.
- Maximum file size: unlimited. The image file uses a part of the available memory for the user application.

#### 7.4.6 Using the visualisation manager



- Familiarise yourself with the following CODESYS functions:  
Visualisation manager → Online Help > CODESYS Visualization > Reference, User Interface > Objects > Object 'Visualization Manager' - 'Settings'

The visualisation manager contains the general settings of the visualisations:

- In the device tree: Double-click on [Application] > [Visualization Manager]
  - ▷ Editor window shows visualisation manager.
- Select the [Properties] tab.
  - ▷ Editor window shows general settings of the visualisations.
- Set the parameters as required.



If the key mapping function is to be used, then the standard keyboard operation must be enabled.

- Enable the checkbox [Activate standard keyboard handling] in the section [Additional settings].

- Save the project to adopt all changes.

#### Defining functions for operating elements



- Familiarise yourself with the following CODESYS functions:  
Keyboard configuration: → Online Help > CODESYS Visualization > Reference, User Interface > Objects > Tab 'Visualization Manager' - 'Default Hotkeys'

The CODESYS function "Standard keyboard shortcuts" allows the user to assign certain functions to the operating elements of the unit (e.g. page change, switch variable). The keyboard functions defined in this way apply to all visualisations in the project.

To assign specific functions to the controls on the unit:

- Start visualisation manager
  - ▷ Editor window shows visualisation manager.
- In the editor window: Select the [Default Hotkeys] tab.
  - ▷ Editor window shows current configuration of the keyboard functions.
- Set the following values:
  - 1st column [Key]: Select the required symbol name. Symbol names of the operating elements (→ 45)
  - 2nd column [Key down]: Define ON level
  - 3rd column [Action type]: Select action type
  - 4th column [Action]: Select action.

- ▶ Define further keyboard functions if required.
- ▶ Save the project to adopt all changes.
- ▷ The standard keyboard shortcuts are defined.

## Global switching of the visualisation



- ▶ Familiarise yourself with the following CODESYS functions:  
Use CurrentVisu variable: → Online Help > CODESYS Visualization > Reference, User Interface > Objects > Object 'Visualization Manager' - 'Settings'

The following describes the global switching of the visualisation with the help of the global CurrentVisu variable `VisuElems.CurrentVisu`.

### Properties of the CurrentVisu variable:

- Data type `String`
- Contains the name of the currently displayed visualisation at runtime of the application.
- The value can be read and written.
- By writing the value in the application, a global switching of the current visualisation takes place on all visualisation devices simultaneously (TargetVisu and WebVisu).

### How to proceed:

- ▶ Enable the CurrentVisu variable in the settings of the visualisation manager under [Settings] > [General Settings].
- ▶ The CurrentVisu variable can be used for global switching of the visualisation.
- ▶ In the application, assign a new value to the CurrentVisu variable, example:  
`VisuElems.CurrentVisu := 'visu1';`
- ▷ The visualisation is switched globally on all visualisation devices (TargetVisu and WebVisu) simultaneously.

## Local switching of the visualisation



- ▶ Familiarise yourself with the following CODESYS functions:  
Use CurrentVisu variable: → Online Help > CODESYS Visualization > Reference, User Interface > Objects > Object 'Visualization Manager' - 'Settings'

The following describes the local switching of the visualisation with the help of buttons/function keys in the visualisation.

Local means that the visualisation switch is only executed on the respective device on which the button was pressed.

### Creating and configuring buttons:

- ▶ Disable the global CurrentVisu variable `VisuElems.CurrentVisu` in the settings of the visualisation manager under [Settings] > [General Settings].
  - ▷ The global switching of the visualisation is disabled.
- ▶ Create 2 visualisations, e.g. `visu1` and `visu2` Creating a visualisation (→ 57)
- ▶ Double-click to open `visu1` / `visu2` in the editor.
- ▶ Drag and drop a button from the [Visualization Toolbox] window onto the visualisation.
- ▶ Highlight the button.
  - ▷ The properties of the button are displayed.

- ▶ Click on [Properties] > [Input configuration] > [OnMouseClicked] > [Configure...].
  - ▷ The window [Input configuration] appears.
- ▶ Highlight [Change Shown Visualization] and click [>].
- ▶ Select visu2 / visu1 with [Selection] > [Assign] > [...].
- ▶ Click on [OK].
  - ▷ The visualisation change is created.
  - ▷ During runtime, the buttons are used to switch from visu1 to visu2 and vice versa.

### Configuring the function key for the button:

- ✓ The visualisations with visualisation switching have been created.
- ▶ Open the visualisations with a double click in the editor.
- ▶ Highlight the button for the visualisation change.
  - ▷ The properties of the button are displayed.
- ▶ Select the required function key under [Properties] > [Input configuration] > [Hotkey] > [Taste]. The function key must be configured in the visualisation manager in the tab [Default Hotkeys].
- ▶ Under [Properties] > [Input configuration] > [Hotkeys] > [Events], set [MouseDown/MouseUp].
  - ▷ The function key is configured as keyboard shortcuts for the button.
  - ▷ During runtime, a switchover from visu1 to visu2 and vice versa can additionally be triggered with the function keys of the unit.

### Configuring the target visualisation

To change the properties of the created visualisation:

- ▶ In the device tree: Double-click on [VisualizationManager] > [TargetVisu]
  - ▷ Editor window shows properties of the target visualisation
- ▶ Set the following values:
  1. [Start visualization]: Select the required visualisation.
  2. [Update rate ms:]: 150
  3. [Scaling options]: Fixed
  4. [Antialiased drawing]: active
  5. [Default Text Input]: Select requested input device.
- ▶ Save the project to adopt all changes.



- ▶ Observe the notes about the configuration of the visualisation task! Configuring a visualisation task (→ 70)

### Configure web visualisation

To change the properties of the created visualisation:

- ▶ In the device tree: Double click on [VisualizationManager] > [WebVisualization]
  - ▷ The editor window shows attributes of the web visualisation
- ▶ Set the required value.
- ▶ Save the project to adopt all changes.



- ▶ Observe the notes about the configuration of the visualisation task! Configuring a visualisation task (→ 70)



## 7.4.7 Creating a visualisation

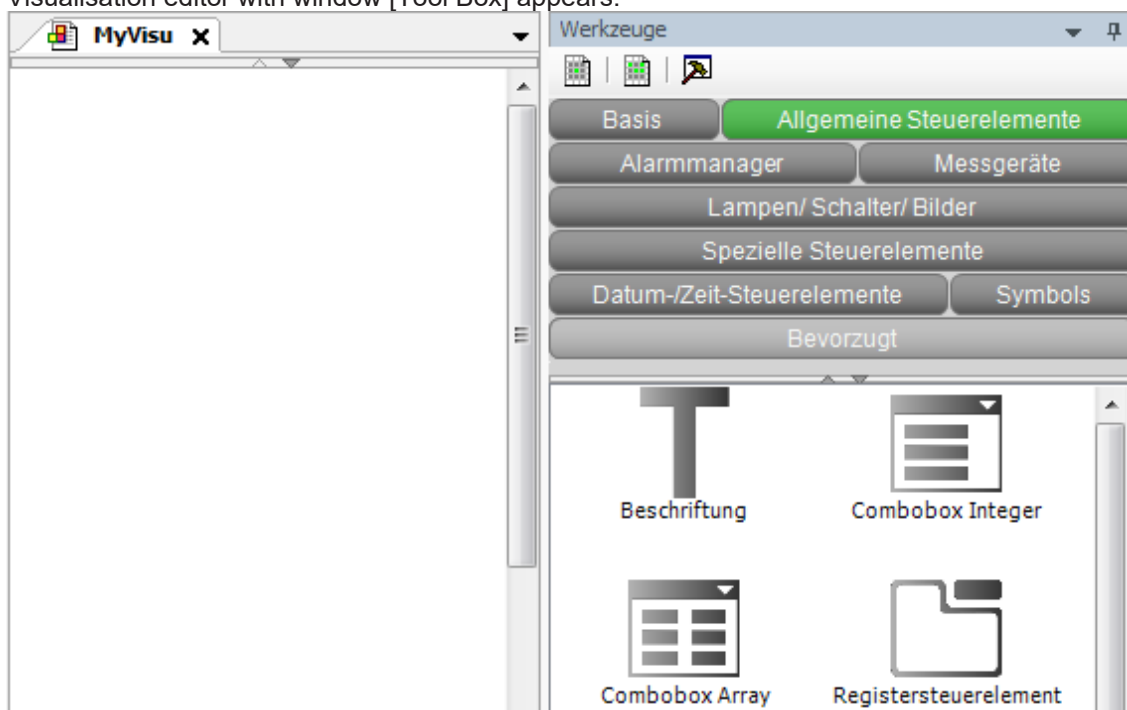


- Familiarise yourself with the following CODESYS functions:  
Visualisation editor: → Online help > CODESYS Visualization > Visualization Editor

Visualisation elements: → Online help > CODESYS Visualization > Designing a Visualization with Elements

To create a visualisation for a PLC application:

- In the device tree: Double-click on [Visualization]
- ▷ Visualisation editor with window [Tool Box] appears.



- Create requested visualisation.
- Save the project to adopt all changes.

## 7.5 Using touch screen functions

### 7.5.1 Notes



- Observe the following information when programming touch screen functions:
  - The device has multitouch functionality.
  - The use of touch operating elements is not suitable for the control of critical functions (e.g. motor start/stop).
- Use the mechanical keys for the implementation of critical functions!
- The touch screen provides no mechanical feedback when a graphical operating element has been activated (e.g. button). Therefore, it may happen that the operator does not know whether the entry was successful.

- ▶ Use visual feedback for graphical operating elements.
- Unfavourable light incidence and soiling of the screen surface might affect the perceptibility of the graphical operating elements.
- ▶ Touch operating elements must be sufficiently dimensioned!
- ▶ Use a clearly readable font in a sufficient font size to label the touch operating elements.
- The touch screen of the device is calibrated at the factory.
- FB DisableTouchScreen (→ [□ 175](#)): The FB activates/deactivates the touch screen functionality of the display. The touch screen is enabled by default when the device will be rebooted.
- FB SetTouchOptimisationMode (→ [□ 177](#)): The function block enables the optimisation mode for a certain operating condition. The user can choose between the following operating conditions:
  - Standard operation
  - Operation with gloves
  - Operation in rain / splash water

## 7.5.2 Configuring input objects of the visualisation



- ▶ Familiarise yourself with the following CODESYS functions:  
Input configuration → Online Help > CODESYS Visualization > Configuring User Inputs > Configuring User Inputs for Visualization Elements

To configure the objects of a visualisation for input via touch screen:

- ▶ Create input object (e.g. button)
- ▶ Mark created input object.
  - ▷ The [Properties] window shows the properties of the input object.
- ▶ Set the following parameters under [Input configuration]:
- ▶ Save the project to adopt all changes.

## 7.5.3 Using multitouch functionality

The device can be used as a multitouch display. For this purpose, the corresponding option must be activated in the CODESYS visualisation manager. The multitouch functionality must be programmed via the CODESYS application.

A corresponding program example can be found in the CODESYS store or ordered from the ifm support.

It is recommended that multitouch applications will only be programmed by experienced users since some standard CODESYS touch entries will no longer be available in multitouch mode.

## 7.6 Operation without touch functionality

For devices without touch functionality (→ Technical data):

- ▶ Select the operating elements (buttons, input fields, etc.) in the visualisation with the navigation keys of the rocker switch. Navigation keys (→ [□ 25](#))
  - ▷ A frame indicates the focused operating element.
- ▶ Enable the operating element with the [RETURN] key of the navigation key element.
  - ▷ The action assigned to the button will be executed.

## 7.7 Using mobile cameras

All device variants support the operation of Ethernet cameras.

Some device variants support the operation of analogue cameras. → Technical data

The image data of a connected camera is streamed into a separate window in the visualisation.

Interface	signal type	Properties
Analogue video input (FBAS)	Analogue	<ul style="list-style-type: none"> <li>• Number of analogue connections→ Data sheet</li> <li>• Automatic NTSC / PAL recognition</li> <li>• All camera images can be visualised simultaneously</li> <li>• Changing between all cameras is possible</li> <li>• Camera image and visualisation representable at the same time (overlaid)</li> <li>• Camera failures will be signalled when the device is switched on</li> <li>• Cameras can be activated / deactivated separately</li> <li>• Rotation of the camera image by 90° / 180° / 270° is possible during operation</li> <li>• Scaling and mirroring of the camera image is possible during operation</li> <li>• Creating screenshots of the camera image</li> </ul>
Ethernet / IP interface	Digital	<ul style="list-style-type: none"> <li>• The number of cameras to be included depends on the corresponding device and the programmed application, e.g.: CR1077 with minor application: Several high-resolution camera images are supported; CR1058 with extensive application: A low-resolution camera image is supported.</li> <li>• Supported CODECS: RTSP and RTP</li> <li>• All camera images can be visualised simultaneously</li> <li>• Changing between all cameras is possible</li> <li>• Camera image and visualisation representable at the same time (overlaid)</li> <li>• Camera failures will be signalled when the device is switched on</li> <li>• Cameras can be activated / deactivated separately</li> <li>• Rotation of the camera image by 90° / 180° / 270° is possible during operation</li> <li>• Scaling and mirroring of the camera image is possible during operation</li> <li>• Creating screenshots of the camera image</li> </ul>

### 7.7.1 Supported cameras

	Analogue cameras	Ethernet cameras
<b>Recommended reference cameras</b>	ifm electronic: O2M200, O2M201, O2M202, O2M203	ifm electronic: Article number ZB086x, e.g. ZB0861, ZB0862, ZB0863, ZB0864
<b>Cameras from other manufacturers</b>	<ul style="list-style-type: none"> <li>• Motec: MC3100-3R (tested)</li> <li>• Vision Techniques: VT70 (tested)</li> </ul>	<ul style="list-style-type: none"> <li>• Motec: MCDE3100 (tested)</li> <li>• Axis: F41 Main Unit / F1035-E Sensor Unit 12m (tested)</li> </ul> <p>The following standards are supported:</p> <ul style="list-style-type: none"> <li>• Codecs: H264 MJPEG</li> <li>• Protocols: RTP (Automotive Industry) Real Time Protocol, RTSP (Webcam) Real Time Streaming Protoco</li> </ul>



► Only use recommended or tested cameras! Using other cameras on the device is without warranty!

### 7.7.2 Configuring and controlling the analogue camera

► Connect the analogue camera with the device.

- Use the following function block to configure and control an analogue camera: FB AnalogueCameraWindowControl (→ 193)
- Configure the camera FB as follows:
  - Activate the camera with `xEnable = TRUE` and deactivate with `xEnable = FALSE`.
  - Configure the size and position of the camera window at the FB input `stWindowControls`.
  - Set the video stream at the FB input `eCamera`.
  - Configure properties and control commands for the camera at the FB input `stCameraControls`.

### Image resolutions for analogue cameras

The following image resolutions are provided by analogue cameras depending on the colour coding system used:

- PAL: 720 x 576 (aspect ratio: 5:4)
- NTSC: 720 x 480 (aspect ratio: 3.2)

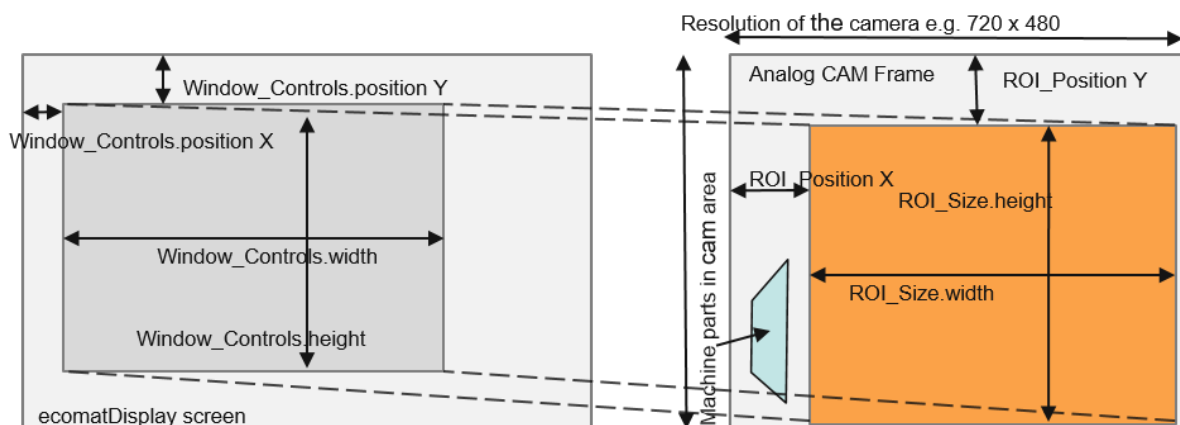
### 7.7.3 Configuring and controlling an Ethernet camera

- Connect the Ethernet camera and the device to a common network.
- Use the following function block to configure and control an Ethernet camera: FB IPCameraWindowControl (→ 195)
- Configure the camera FB as follows:
  - Activate the camera with `xEnable = TRUE` and deactivate with `xEnable = FALSE`.
  - Configure the size and position of the camera window at the FB input `stWindowControls`.
  - Specify a unique camera index at the FB input `usiCameraIndex`.
  - Configure properties and control commands for the camera at the FB input `stCameraControls`.

### 7.7.4 Configuring the Region of Interest (ROI)

In addition to displaying the entire camera image, it is possible to show any image section at any position and in any size on the display. The image section is called Region of Interest (ROI). This technology is designed to hide interfering objects, such as machine parts. If the size of the ROI is identical with the resolution of the camera image, the complete camera image will be shown on the display.

The following graphic demonstrates the correspondences:



Window_Controls	Object with size and position of the camera window on the display.
ROI	Object and size and position of the camera image section.

## 7.8 Using a PDF viewer

► Display the PDF file in the PDF viewer on the device:

1. Transfer the PDF file to the device, e.g. with the CODESYS function [Add Object / External File]. Integrating external files (→ 51)
2. Use the following function block to configure and display a PDF in a PDF viewer: FB PDF\_Viewer PDF\_Viewer (→ 198)

► Configure the FB PDF\_Viewer as follows:

1. Specify the path and name of the PDF file at the FB input sFileName.
2. Configure the size and position of the PDF viewer window at the FB input stWindowControls.
3. Configure properties and control commands for the PDF viewer at the FB input iq\_stPdfControls.

### 7.8.1 Example

Programming example for the PDF viewer:

✓ A PDF file already exists on the device. Integrating external files (→ 51)

► Specify the path of the PDF file at the FB input sFileName.

Default path for file transfer via CODESYS:

/home/cds-apps/PlcLogic/Application/filename.pdf



► Observe the conventions for the file name. File name conventions (→ 51)

► If necessary, check under [Device > Files] whether the file is available on the device.

► If necessary, check the spelling and the upper/lower case of the file under [Device > Files].

► Define the structure for window properties and PDF display:

VAR

stPDF\_Window: stWINDOW\_CONFIG;

stPDF\_ctrl: stPDF\_CONTROLS;

END\_VAR

► Define window properties for PDF viewer:

stPDF\_Window.stPosition.uiX:=10; (\* X position upper left corner \*)

stPDF\_Window.stPosition.uiY:=10; (\* y position upper left corner \*)

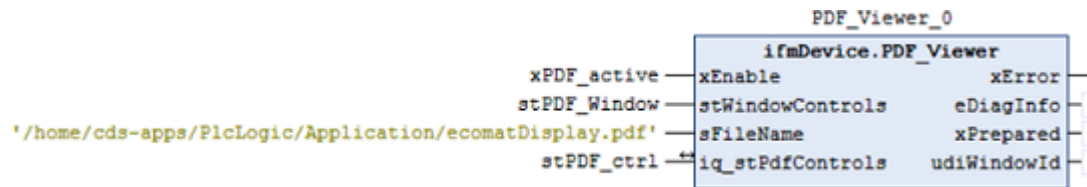
stPDF\_Window.stSize.uiHeight:=450; (\* height of pdf viewer window \*)

stPDF\_Window.stSize.uiWidth:=600; (\* width of pdf viewer window \*)

► Define properties for PDF display:

stPDF\_ctrl.xSetFullScreen:=TRUE; (\* full screen mode on \*)

Call the FB PDF viewer with the required parameters:



- ▶ Use full screen mode.
- ▷ Scrolling and zooming via the touch function causes a high CPU load.

## 7.9 CSV file logging

The device offers the possibility to write and read CSV files, e.g. to log machine data.

### 7.9.1 Writing a CSV file

To write a CSV file, the following POU's are available:

Name	Description	Reference
WriteCSVHeader	Write header line in CSV file	WriteCSVHeader (→ 249)
WriteCSVData_Linear	Write data in linear mode to a CSV file.	WriteCSVData_Linear (→ 243)
WriteCSVData_Ring	Write data in ring mode to a CSV file.	WriteCSVData_Ring (→ 246)

Useful auxiliary functions:

Name	Description	Reference
ANY_TYPE_TO_STRING	Converts any data type into a string.	ANY_TYPE_TO_STRING (→ 264)
ifmCONCAT	Combines two strings of up to 1000 characters each into one string of up to 1000 characters.	ifmCONCAT (→ 265)
ifmFIND	Searches a string of up to 1000 characters.	ifmFIND (→ 266)
ifmMID	Outputs a substring of up to 1000 characters of a string of up to 1000 characters.	ifmMID (→ 267)



The ifm auxiliary functions ifmCONCAT, ifmFIND and ifmMID process strings with a maximum length of 1000 characters. The length of a standard string in CODESYS is 255 characters. String format uiGenericLogSizeMax (→ 63)

Logging of any data type and writing to a CSV file, e.g. REAL, WORD, DWORD, STRING is possible.

Maximum: Logging of up to 1000 characters per operation and up to 10 log operations per second. In total: Up to 10000 characters per second

- ▶ Use system resources wisely: Only save as much data as is necessary for the application.
- ▶ Use an external data carrier, e.g. a USB stick, as storage location for the CSV file.

#### How to proceed:

- ▶ Create the CSV file with the FB `WriteCSVHeader` and write the header line into the CSV file.
- ▶ Convert the variable values into strings with the function `ANY_TYPE_TO_STRING`.
- ▶ Use the `ifmCONCAT` function to compile the data into a string for a CSV file line.

- Write the CSV file line into the CSV file with FB `WriteCSVData_Linear` or FB `WriteCSVData_Ring`.

## 7.9.2 Reading a CSV file

To read a CSV file read, the following POU is available:

Name	Description	Reference
ReadCSVData	Read data from a CSV file written with FB <code>WriteCSVData_Linear</code> or FB <code>WriteCSVData_Ring</code> .	ReadCSVData (→ 236)

## 7.9.3 String format `uiGenericLogSizeMax`

The FBs and functions for writing and reading CSV files (data logging) use the string data type `ifmGCL.uiGenericLogSizeMax` `ifmGCL` (GVL) (→ 269)

The string data type `ifmGCL.uiGenericLogSizeMax` stores strings with a maximum length of 1000 characters.

The standard string data type of CODESYS stores strings with a maximum length of 255 characters.

## 7.10 CODESYS IIoT Libraries SL

### ATTENTION

If the device is operated in an unprotected network environment.

- ▷ Unauthorised read or write access to data is possible.
- ▷ Unauthorised manipulation of the device function is possible.
- Check and restrict access options to the device.
- Restrict access to authorised users.
- Choose a secure method to connect with the device (e.g. VPN).
- Use encrypted data transmission (e.g. https / TLS).

With the purchase of the device from firmware V2, the user simultaneously acquires a valid licence for the use of the `IIoT Libraries SL` software package from CODESYS.

The licence is on the device when it is delivered.

The `IIoT Libraries SL` software package can be downloaded from the CODESYS website.

The `IIoT Libraries SL`, for example, offer the following features:

- Easy implementation of web services on the device
- Easy connection to servers or web interfaces such as NodeRED
- http post / http get
- MQTT Client: Open Source IoT Protocol
- AWS IoT Core Client
- Azure IoT Hub Client
- JSON Utilities
- XML Utilities
- Mail Service: easy sending and receiving of emails

## 7.11 Using CANopen



- Observe the notes on task configuration. Configuring task processing (→ 69)

The following POU's are available to access a CAN interface configured for CANopen operation in an application.

- Before: Configure the unit as CANopen Manager (Master). CANopen: configuring CANopen Manager (master) (→ 40)

### 7.11.1 CANopen: Sending and receiving SDO

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

Name	Description	Reference
COP_SDRead	Read Service Data Object (SDO)	COP_SDRead (→ 80)
COP_SDOwrite	Write Service Data Object (SDO)	COP_SDOwrite (→ 82)

### 7.11.2 CANopen: Network Management (NMT)

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

Name	Description	Reference
COP_GetNodeState	Request state of one or several CANopen devices	COP_GetNodeState (→ 78)
COP_SendNMT	Send NMT control command to a CANopen device	COP_SendNMT (→ 84)

## 7.12 Using RawCAN (CAN Layer 2)



- Observe the notes on task configuration. Configuring task processing (→ 69)

The following POU's are available to access a CAN interface configured for CANopen operation in an application.

- Before: Configure the CAN interface for operation as RawCAN (CAN Layer 2). RawCAN: configuring CANLayer 2 (→ 39)

### 7.12.1 RawCAN: Controlling CAN network nodes

The following POU's are available to control a node in a CAN network:

Name	Description	Reference
CAN_Enable	Activate CAN node	CAN_Enable (→ 271)
CAN_Recover	Reboot CAN node	CAN_Recover (→ 273)

### 7.12.2 RawCAN: Sending and receiving CAN messages

The following POU's are available to send or receive messages in a CAN network:

Name	Description	Reference
CAN_Rx	Receive CAN message	CAN_Rx (→ 279)



Name	Description	Reference
CAN_RxMask	Receive CAN messages	CAN_RxMask (→ □ 281)
CAN_RxRange	Receive CAN messages for standard and extended frames	CAN_RxRange (→ □ 283)
CAN_RxRange	Receive CAN messages for extended frames	CAN_RxRangeExt (→ □ 285)
CAN_Tx	Send CAN message	CAN_Tx (→ □ 289)

### 7.12.3 RawCAN: Requesting and sending remote CAN messages

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

Name	Description	Reference
CAN_RemoteRequest	Send request for a remote message	CAN_RemoteRequest (→ □ 275)
CAN_RemoteResponse	Reply to the request of a remote message	CAN_RemoteResponse (→ □ 277)

## 7.13 Using J1939

To use the SAE J1939 network protocol, CODESYS GmbH provides the library IoDrvJ1939 with additional functions.

Set the configuration via the unit tree as follows:

### 7.13.1 Adding a CAN bus

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

### 7.13.2 Assign CAN interface

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

### 7.13.3 Adding J1939 Manager

- ▶ In the CODESYS device tree: Right-click on [Communication] > [CAN] > [ifmCANBus].
- ▶ Select [Add Device...].
  - ▷ The window [Add Device] appears.

- ▶ In the [Device] section: Vendor: select <All vendors>.
- ▶ In the list below: Select [Field buses] > [SAE J1939] > [J1939 Manager] > [J1939\_Manager].
- ▶ Confirm the selection with [Add Device].
- ▶ Close the window [Add Device] with the [Close] button.

#### 7.13.4 Setting 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.

#### 7.13.5 Attaching J1939-ECU

- ▶ In the CODESYS device tree: Right-click on [Communication] > [CAN] > [ifmCANBus] > [J1939\_Manager].
- ▶ Select [Add Device...].
  - ▷ The window [Add Device] appears.
- ▶ In the [Device] section: Make: select <All vendors>.
- ▶ In the list below: Select [Fieldbuses] > [J1939] > [J1939\_ECU].
- ▶ Confirm the selection with [Add Device].
- ▶ Close the window [Add Device] with the [Close] button.

#### 7.13.6 Setting 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 th section [General] according to the specific application:

Specific application:	[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].

## 7.14 Using EtherNet/IP



- ▶ Familiarise yourself with the following CODESYS functions:  
EtherNet/IP adapter: → Online Help > Fieldbus Support > EtherNet/IP Configurator > EtherNet/IP Adapter

EtherNet/IP scanner: → Online Help > Fieldbus Support > EtherNet/IP Configurator > EtherNet/IP Scanner

- ▶ To use the EtherNet/IP network protocol, set the configuration via the device tree as follows:

### 7.14.1 Adding an Ethernet adapter to an Ethernet

- ▶ In the CODESYS device tree: Right-click on [Communication] > [Ethernet].
- ▶ Select [Add Device...].
  - ▷ The window [Add Device] appears.
- ▶ Select <All vendors> in the [Vendor] list.
- ▶ In the list below, select [Fieldbuses] > [EtherNet/IP] > [Ethernet Adapter] > [Ethernet].
- ▶ Conform the selection with [Add Device]
- ▶ Close the window [Add Device] with the [Close] button.
- ▷ The Ethernet adapter is attached.

### 7.14.2 Attaching an EtherNet/IP adapter

- ▶ In the CODESYS device tree: Right-click on [Communication] > [Ethernet] > [Ethernet\_1].
- ▶ Select [Add Device...].
  - ▷ The window [Add Device] appears.
- ▶ Select <All vendors> in the [Vendor] list.
- ▶ In the list below, select [Fieldbuses] > [EtherNet/IP] > [EtherNet/IP Local Adapter] > [EtherNet/IP Adapter].
- ▶ Conform the selection with [Add Device]
- ▶ Close the window [Add Device] with the [Close] button.
- ▷ The EtherNet/IP adapter is attached.

### 7.14.3 Attaching an EtherNet/IP module

- ▶ In the CODESYS device tree: Right-click on [Communication] > [Ethernet] > [Ethernet\_1] > [Ethernet\_IP\_Adapter].
- ▶ Select [Add Device...].
  - ▷ The window [Add Device] appears.
- ▶ Select <All vendors> in the [Vendor] list.
- ▶ In the list below, select [Fieldbuses] > [EtherNet/IP] > [EtherNet/IP Module] > [EtherNet/IP Module].
- ▶ Conform the selection with [Add Device]
- ▶ Close the window [Add Device] with the [Close] button.
- ▷ The EtherNet/IP module is attached.

---

#### 7.14.4 Configuring the EtherNet/IP interface

- ▶ In the CODESYS device tree, double-click [Communication] > [Ethernet] > [Ethernet\_1].
- ▶ Configure the interface as required, e.g.:
  - IP address
  - Ethernet Device I/O Image
  - Ethernet Device IEC objects

### 7.15 Using Modbus



- ▶ Familiarise yourself with the following CODESYS functions:  
Modbus master: → Online Help > Fieldbus Support > Modbus Configurator > Modbus Master

Modbus slave device: → Online Help > Fieldbus Support > Modbus Configurator > Modbus Slave Device

To use the Modbus network protocol, set the configuration via the device tree as follows:

#### 7.15.1 Adding an Ethernet adapter to an Ethernet

- ▶ In the CODESYS device tree: Right-click on [Communication] > [Ethernet].
- ▶ Select [Add Device...].
  - ▷ The window [Add Device] appears.
- ▶ Select <All vendors> in the [Vendor] list.
- ▶ In the list below, select [Fieldbuses] > [Ethernet Adapter] > Ethernet [Adapter] > [Ethernet].
- ▶ Conform the selection with [Add Device]
- ▶ Close the window [Add Device] with the [Close] button.
- ▷ The Ethernet adapter is attached.

#### 7.15.2 Attaching a Modbus TCP master

- ▶ In the CODESYS device tree: Right-click on [Communication] > [Ethernet] > [Ethernet\_1].
- ▶ Select [Add Device...].
  - ▷ The window [Add Device] appears.
- ▶ Select <All vendors> in the [Vendor] list.
- ▶ In the list below, select [Fieldbuses] > [Modbus] > Modbus [TCP Master] > Modbus [TCP Master].
- ▶ Conform the selection with [Add Device]
- ▶ Close the window [Add Device] with the [Close] button.
- ▷ The TCP master mode is attached.

#### 7.15.3 Attaching a Modbus TCP slave device

- ▶ In the CODESYS device tree: Right-click on [Communication] > [Ethernet] > [Ethernet\_1] > [Modbus\_TCP\_Master].
- ▶ Select [Add Device...].
  - ▷ The window [Add Device] appears.
- ▶ Select <All vendors> in the [Vendor] list.

- ▶ In the list below, select [Fieldbuses] > [Modbus] > [Modbus TCP Slave Device] > Modbus [TCP Slave Device.]
- ▶ Conform the selection with [Add Device]
- ▶ Close the window [Add Device] with the [Close] button.
- ▷ The TCP Slave Device mode is added to the Modbus TCP Master.

#### 7.15.4 Configuring a Modbus TCP slave device

- ▶ In the CODESYS device tree: Double-click on [Communication] > [Ethernet] > [Ethernet\_1] > [Modbus\_TCP\_Master] > [Modbus\_TCP\_Slave].
  - ▷ The configuration window for the Modbus\_TCP\_Slave appears.
- ▶ In the following tabs, set the parameters as required:
  - [General]: Enter the IP address of the Modbus\_TCP\_Slave. Optional: Enter [Response Timeout] and [Port].
  - [Modbus Slave Channel]: Add the required Modbus channels (Read Holding Registers).
  - [Modbus TCPSlave I/O Mapping]: Define variables and assign the values of the Modbus registers to them.



- ▶ For more information about the Modbus settings, the Modbus communication and the register configuration of the respective Modbus slave: → Operating instructions of the Modbus slave.
- ▶ After completing the Modbus configuration, evaluate the variables in the application program.

### 7.16 Configuring task processing



- ▶ Familiarise yourself with the following CODESYS functions:  
Task configuration: → Online Help > CODESYS Development System > Programming of Applications > Task Configuration

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

CODESYS automatically generates the following tasks when creating the project and visualisation using the ecomatDisplay template:

Name	Description	Reference
Task	Task for the processing of the main program [PLC_PRG (PRG)]	Configuring a task (→ □ 70)
VISU_TASK	Task for the processing of the visualisations	Configuring a visualisation task (→ □ 70)



- For subprograms with POU's to be executed several times per PLC cycle:
  - ▶ Create new task.
  - ▶ Configure task properties:
    - [Type]: Cyclic
    - [Interval]: Requested cycle time
  - ▶ Assign the sub-program with POU's to the newly created task.



If the CAN buses are heavily utilised:

- ▶ Create an individual task for each CAN interface.
- ▶ Configure task properties:
  - [Priority]: High (< 5)
  - [Type]: Cyclic
  - [Interval]: requested cycle time (=transmission interval)
- ▶ Assign sub-programs with the POU's for CAN communication to the CAN tasks.

### 7.16.1 Configuring a task

The task determines the execution of the standard program [PLC\_PRG]. The programmer can assign additional subprograms to the task.

To set the properties of the task:

- ▶ In the device tree: Double-click on [Task]
  - ▷ In the editor window: The [Configuration] tab shows the current configuration of the task.
- ▶ Set the following values:
  - [Priority (0...31)]: 1
  - [Type]: Cyclic
  - [Interval]: t#10ms
- ▶ Save the project to adopt all changes.

### 7.16.2 Configuring a visualisation task

Each visualisation is executed separately from the program code in a separate task.

To set the attributes of the visualisation task:

- ▶ In the device tree: Double-click on [VISU\_TASK]
  - ▷ In the editor window: Tab [VISU\_TASK] > [Configuration] shows current configuration of the visualisation task.
- ▶ Set the following values:
  - [Priority (0...31)]: <16
  - [Type]: Cyclic
  - [Interval]: t#20ms



- ▶ Assign a priority that is as low as possible to the visualisation task ([VISU\_TASK]) to avoid interruption of other tasks that are important for the core functions of the application.
- ▶ Execute the VISU\_TASK in appropriate cyclic intervals to save the resources of the device-internal PLC and the fieldbus network.

- ▶ Save the project to adopt all changes.

## 8 Operation

### 8.1 Transferring CODESYS project to the device



- ▶ Familiarise yourself with the following CODESYS functions:  
Transfer application to the device → Online help > CODESYS Development System > Transfer application to the PLC  
  
Monitoring → Online help > CODESYS Development System > Application during the runtime > Monitoring of values
- ▶ Observe the notes about the states of the PLC application! Operating states of the PLC application (→ [72](#))

GB

#### 8.1.1 Loading an application to ecomatDisplay

To load the created application to the device and store it non-volatilely:

- ✓ Connection between PC/laptop and ecomatDisplay is established.
- ✓ Communication path has been set. Configuring the programming interface (→ [19](#))
- ✓ Project has been tested.
- ▶ In the project tree: Click on [Application].
- ▶ Select [Build] > [Build].
  - ▷ CODESYS creates program code of the application.
- ▶ Select [Online] > [Login].
  - ▷ CODESYS changes to the online mode.
  - ▷ CODESYS loads active application to the device (download).
  - ▷ Application on the device is in the STOP state.
- ▶ Select [Debug] > [Start].
  - ▷ Application on the device changes to the RUN state.
- ▶ Select [Online] > [Create Boot Application].
- ▷ CODESYS stores the application non-volatilely on the device.

#### 8.1.2 Deleting an application from the device

To delete an application stored on the device:

- ▶ In the device tree: Click on [Application].
- ▶ Select [Online] > [Login].
  - ▷ CODESYS changes to the online mode.
- ▶ In the device tree: Double-click [Device (ecomatDisplay)]
- ▶ Editor window shows device settings.
- ▶ Select the [Applications] tab.
- ▶ Click on [Refresh List].
  - ▷ Editor window shows the applications stored on the device.
- ▶ Click on [Remove all] to delete all applications.
  - or -
  - Select the required application and click on [Remove] to delete individual applications.
- ▷ CODESYS deletes the selected applications from the device.

---

## 8.2 Operating states of the PLC application

The applications stored on the ecomatDisplay are executed independently from each other in separate tasks. An application can have the following operating states:

- **Unload**  
No application is stored on the device.
- **RUN**
  - The application is stored on the device.
  - The application is processed cyclically.
- **STOP**
  - The application is stored on the device.
  - The application is not processed.

### 8.2.1 Displaying operating mode of the PLC application

To display the current operating status of the PLC application stored on the device:

- ▶ In the device tree: Symbol [Application] shows the current status
  - or -
- ▶ In the online mode: CODESYS status bar shows the current state of the application.

### 8.2.2 Starting the PLC application

To start the execution of the PLC application:

- ▶ In the device tree: Right-click on [Application] and select [Set Active Application].
- ▶ Select [Online] > [Login].
  - ▷ CODESYS changes to the online mode.
- ▶ Select [Debug] > [Start].
  - ▷ Application changes to the RUN state.
- ▶ Optional: Repeat process for further applications.

### 8.2.3 Stopping the PLC application

To stop the execution of the PLC application:

- ▶ In the device tree: Right-click on Application and select [Set Active Application].
- ▶ Select [Online] > [Login].
  - ▷ CODESYS changes to the online mode.
- ▶ Select [Debug] > [Stop].
  - ▷ Application changes to the STOP state.
- ▶ Optional: Repeat process for further applications.

## 8.3 Reset

### 8.3.1 Supported reset variants

The following table shows the reset variants supported by the device-internal PLC and the resulting system behaviour:



Type of reset	System behaviour	Triggering actions
<b>Reset (warm)</b>	<ul style="list-style-type: none"> <li>The application passes into the STOP mode.</li> <li>Standard variables (VAR) of the applications are reinitialised.</li> <li>Remanent variables (VAR RETAIN) of the application keep their current values.</li> </ul>	Resetting the application (warm) (→ □ 73)
<b>Reset (cold)</b>	<ul style="list-style-type: none"> <li>The application passes to the STOP state.</li> <li>All variables (VAR, VAR RETAIN) of the application will be re-initialised.</li> </ul>	Resetting the application (cold) (→ □ 73)
<b>Reset (default)</b>	<ul style="list-style-type: none"> <li>The application passes into the STOP mode.</li> <li>The application on the PLC will be deleted.</li> <li>All variables (VAR, VAR RETAIN) of the application will be re-initialised.</li> <li>PLC is reset to the original state.</li> </ul>	Resetting the application (origin) (→ □ 73)



A variable declared without an initialisation value is initialised with the variable-specific default value (e.g. `INT = 0`).

### 8.3.2 Resetting the application (warm)

To reset the application:

- In the device tree: Select [Application] and set it as the active application
- Select [Online] > [Login].
  - ▷ 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.

### 8.3.3 Resetting the application (cold)

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

### 8.3.4 Resetting the application (origin)

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
- ▷ PLC is reset to the original state.

## 8.4 Displaying system information

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
[Firmware version]	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 ecomatDisplay.
- ▶ Select [Online] > [Login].
  - ▷ CODESYS changes to the online mode.
- ▶ In the device tree: Double-click on [System\_Info].
- ▶ In the editor window: Select the [Parameter] tab.
- ▷ In the editor window: Table shows current values of the system parameters.

## 9 ifm function libraries

This chapter contains the detailed description of the function libraries provided by ifm electronic for programming the device under CODESYS 3.5.

### 9.1 Behaviour model of the ifm function blocks

This chapter describes the behaviour model of the ifm function blocks for the ecomatDisplay.

#### 9.1.1 General

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 Messages / diagnostic codes of the function blocks (→ 75)	

All inputs and outputs in the function block that belong to the ifm behaviour model are featured at the top.

#### Messages / diagnostic codes of the function blocks

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	Signalled state:	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 (→ 75)

#### 9.1.2 Behaviour model ENABLE

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

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

### 9.1.3 Behaviour model EXECUTE

Function blocks that have the EXECUTE behaviour model are processed once after a rising edge at the xExecute input.

The signal on xExecute must remain set to TRUE until xDone = TRUE or xError = TRUE. If the signal to xExecute becomes FALSE beforehand, the edit process of the function block is aborted without result.

If the function block has executed its function successfully, the output xDone will be set to TRUE for one cycle.

## Response to errors

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

---

## 9.2 Library ifm\_ecomatDisplay\_Cnt

The library is a container library. It contains all function libraries required for the programming of the device.

- ifmCANopenManager (→ [78](#))
- ifmDevice\_ecomatDisplay (→ [86](#))
- ifmFileUtil (→ [220](#))
- ifmRawCAN (→ [271](#))

## 9.3 ifmCANopenManager.library

The library contains program blocks (POU) and data structures for the programming of the functionality of a CANopen Manager.

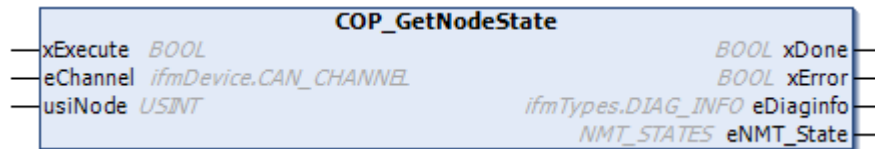
### 9.3.1 COP\_GetNodeState

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmCANopenManager.library

**Symbol in CODESYS:**



#### Description

The FB indicates the current state of a CANopen node.

#### Input parameter

Parameter	Data type	Meaning	Possible values
xExecute	BOOL	Control execution of the FB	<ul style="list-style-type: none"><li>FALSE: Do not execute FB</li><li>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.</li></ul>
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	CAN_CHANNEL (ENUM) (→ 204)
usiNode	USINT	ID of the CANopen node	<ul style="list-style-type: none"><li>0: Local device</li><li>1...127: ID of the CANopen node</li></ul>

#### Output parameter

Parameter	Data type	Meaning	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>An error has 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	→ NMT_STATES (ENUM) (→ S. )	

Diagnostic codes:

- STAT\_INACTIVE Status: FB/Function is inactive.

---

• STAT_BUSY	Status: FB/Function is currently executed.
• STAT_DONE	Status: 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

### 9.3.2 COP\_SDRead

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmCANopenManager.library  
**Symbol in CODESYS:**



#### Description

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 parameter

Parameter	Data type	Meaning	Possible values
xExecute	BOOL	Control execution of the FB	<ul style="list-style-type: none"><li>FALSE: Do not execute FB</li><li>TRUE: Execute FB.</li></ul> 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	CAN_CHANNEL (ENUM) (→ 204)
usiNode	USINT	ID of the CANopen node	<ul style="list-style-type: none"><li>0: Local device</li><li>1...127: ID of the CANopen node</li></ul>
uiIndex	UINT	Index in object directory	
usiSubIndex	USINT	Subindex of the index in the object directory	
pData	Pointer to USINT	Pointer on buffer storage	
udiBuffLen	UDINT	Size of the buffer storage (in byte)	
tTimeout	TIME	Max. response time	e.g.. T#25ms

#### Output parameter

Parameter	Data type	Meaning	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



Parameter	Data type	Meaning	Possible values
xError	BOOL	Indication if an error occurred during the FB execution	TRUE <ul style="list-style-type: none"> <li>An error has 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      Status: FB/Function is inactive.
- STAT\_DONE      Status: 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

### 9.3.3 COP\_SDOWrite

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmCANOpenManager.library  
**Symbol in CODESYS:**



#### Description

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 parameter

Parameter	Data type	Meaning	Possible values
xExecute	BOOL	Control execution of the FB	<ul style="list-style-type: none"> <li>FALSE: Do not execute FB</li> <li>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.</li> </ul>
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	CAN_CHANNEL (ENUM) (→ 204)
usiNode	USINT	ID of the CANOpen node	<ul style="list-style-type: none"> <li>0: Local device</li> <li>1...127: ID of the CANOpen node</li> </ul>
uiIndex	UINT	Index in 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 parameter

Parameter	Data type	Meaning	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>An error has 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	Status: FB/Function is inactive.
• STAT_DONE	Status: 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

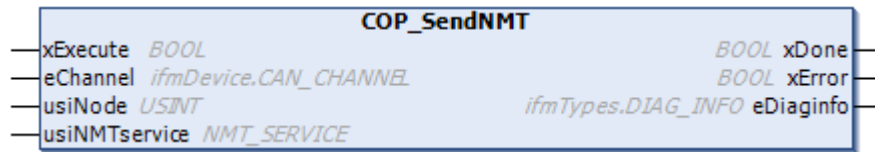
### 9.3.4 COP\_SendNMT

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmCANopenManager.library

**Symbol in CODESYS:**



#### Description

The FB sends a command for the control of a CANopen node.

#### Input parameter

Parameter	Data type	Meaning	Possible values
xExecute	BOOL	Control execution of the FB	<ul style="list-style-type: none"> <li>FALSE: Do not execute FB</li> <li>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.</li> </ul>
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	CAN_CHANNEL (ENUM) (→ □ 204)
usiNode	USINT	ID of the CANopen node	<ul style="list-style-type: none"> <li>0: Local device</li> <li>1...127: ID of the CANopen node</li> </ul>
usiNMTservice	NMT_SERVICE	Command for the control of a CANopen node	NMT_SERVICE (ENUM) (→ □ 85)

#### Output parameter

Parameter	Data type	Meaning	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>An error has 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       | Status: FB/Function is inactive.  |
| • STAT_DONE           | Status: 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!

### 9.3.5 NMT\_SERVICE (ENUM)

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

### 9.3.6 NMT\_STATES (ENUM)

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

## 9.4 Library ifmDevice\_ecomatDisplay.library

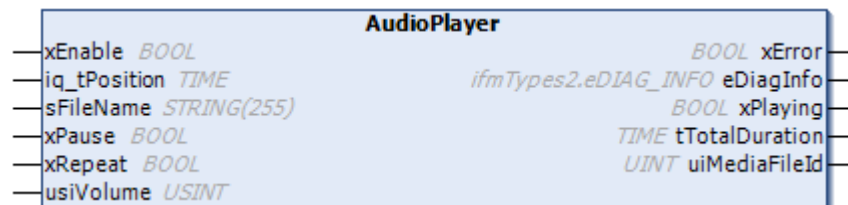
The library contains the following:

- device-specific data structures
- device-specific enumeration types
- device-specific global variables and constants
- device-specific functions

### 9.4.1 Audio

#### AudioPlayer

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



#### Description

The FB enables / disables the audio player of the device. The Audio Player offers the following functions:

- Playing the content of an audio file (supported formats:
- Controlling the play process (pause, repeat, start position)
- Setting the playback volume
- Displaying the runtime of an audio file
- Displaying the activity of the audio player (on, off)

#### Input parameter

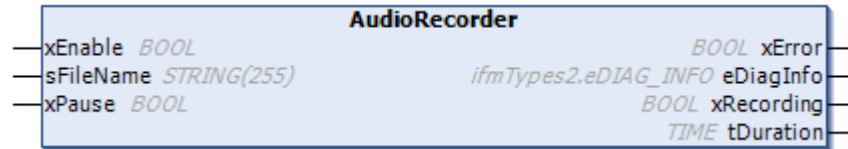
Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
sFileName	STRING(255)	Path and name of the audio file	e.g. \??\??\sound1.wav	
xPause	BOOL	controls the "pause" function of the audio player	FALSE	resume playback and disable mute
			TRUE	play is paused and mute enabled
xRepeat	BOOL	controls the "repeat" function of the audio player	FALSE	repeat disabled
			TRUE	repeat enabled
usiVolume	USINT	playback volume	0...100	

## Output parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xPlaying	BOOL	shows if the audio file is playing	FALSE	audio file is not playing
			TRUE	audio file is playing
tTotalDuration	TIME	Overall length of the audio file (runtime) in seconds.		
uiMediaField	UINT	File system ID of the audio file	0...65536	

## AudioRecorder

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The FB enables / disables the audio recorder of the device. The audio recorder offers the following functions:

- Record the audio signal at the line-in input of the unit and save it as a file (supported format: .wav)
- Controlling the recording process (start/stop, pause / resume)
- Displaying the length of the recording
- Displaying the activity of the audio recorder (active/pause)
- Overwriting the file if it already exists
- Changing the file name during the recording
- Displaying error diagnostics

## Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
sFileName	STRING(255)	Path and name of the audio file	e.g. \??\??\sound1.wav	
xPause	BOOL	controls the "pause" function of the audio recorder	FALSE	Continue recording
			TRUE	recording pauses

## Output parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xRecording	BOOL	shows if the audio file is being recorded	FALSE	The audio file is not being recorded; pause enabled
			TRUE	The audio file is being recorded
tDuration	TIME	Length of the recorded audio file in seconds		

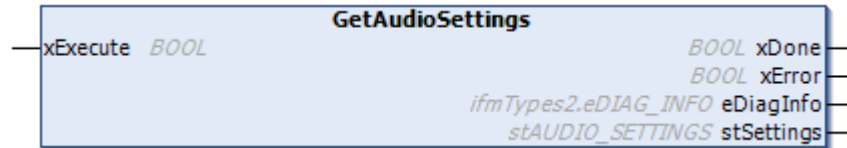


## Diagnostic codes:

• STAT_INACTIVE	Status: FB/Function is inactive.
• STAT_PREPARING	Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• STAT_PAUSED	Status: FB/function has the "pause" status.
• STAT_RECORDING	Status: FB/function has the "recording" status.
• ERR_INVALID_VALUE	Error: At least one input parameter is invalid. Function call has been stopped. Invalid values: <ul style="list-style-type: none"> <li>• The value at sFileName is ZERO (e.g. ")</li> <li>• The value at sFileName ends with "/" (e.g. adirectory)</li> <li>• The value on sFileName contains consecutive slashes (e.g. "////")</li> <li>• The directory or subdirectory does not exist</li> <li>• Audio file format is not supported (only .wav)</li> </ul>
• ERR_MULTIMEDIA_RECORDING_START	<ul style="list-style-type: none"> <li>• Recording start error</li> </ul>
• ERR_MULTIMEDIA_RECORDING_STOP	<ul style="list-style-type: none"> <li>• Recording stop error</li> </ul>
• ERR_MULTIMEDIA_RECORDING_PAUSE	<ul style="list-style-type: none"> <li>• Recording pause error</li> </ul>
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## GetAudioSettings

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The FB reads the currently set volume values of the different audio channels (master, loudspeaker L + R, headphones L + R, Line-In) and provides the values.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

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 parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
stSettings	stAUDIO_SETTINGS	Volume settings of the device's audio channels.	→ stAUDIO_SETTINGS (STRUCT)	

## Diagnostic codes:

- STAT\_INACTIVE
- STAT\_PREPARING
- STAT\_DONE

Status: FB/Function is inactive.

Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.

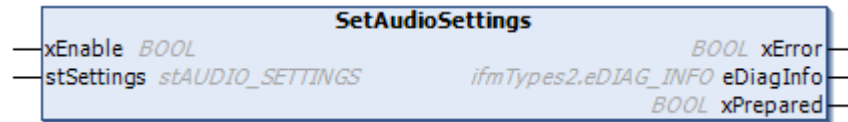
Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.

---

• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!
• ERR_MULTIMEDIA_GET_MASTER_VOLUME	Error while reading the master audio volume
• ERR_MULTIMEDIA_GET_SPEAKER_VOLUME	Error while reading the loudspeaker volume
• ERR_MULTIMEDIA_GET_HEADPHONE_VOLUME	Error while reading the headphone volume
• ERR_MULTIMEDIA_GET_LINEIN_VOLUME	Error while reading the LineIn volume
• ERR_MULTIMEDIA_GET_RECORDING_VOLUME	Error while reading the recording volume

## SetAudioSettings

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block configures the volume values of the various audio channels (master, loudspeaker L + R, headphones L + R, Line-In) of the device. The lower-level system functions will only be executed if the value is changed.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

- Only call up one instance of the function block within the application!

## Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
stSettings	stAUDIO_SETTINGS	Volume settings of the device's audio channels.	→ stAUDIO_SETTINGS (STRUCT)	

## Output parameter

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>An error has 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 outputs are invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed

## Diagnostic codes:

- |  |   |
|--|---|
| <ul style="list-style-type: none"> <li>STAT_INACTIVE</li> </ul>      | Status: FB/Function is inactive.  |
| <ul style="list-style-type: none"> <li>STAT_PREPARING</li> </ul>     | Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle. |
| <ul style="list-style-type: none"> <li>STAT_DONE</li> </ul>          | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.           |
| <ul style="list-style-type: none"> <li>ERR_INSTANCE_LIMIT</li> </ul> | Error: More than one instance of the FB has been created; this FB instance is not executed                          |

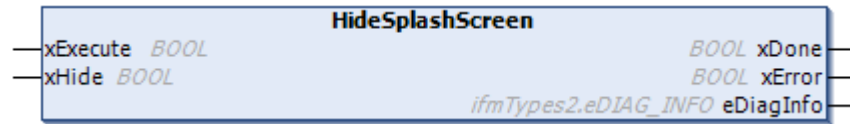
---

• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!
• ERR_MULTIMEDIA_SET_MASTER_VOLUME	Error while setting the master audio volume
• ERR_MULTIMEDIA_SET_SPEAKER_VOLUME	Error while setting the loudspeaker volume
• ERR_MULTIMEDIA_SET_HEADPHONE_VOLUME	Error while setting the headphone volume
• ERR_MULTIMEDIA_SET_LINEIN_VOLUME	Error while setting the LineIn volume
• ERR_MULTIMEDIA_SET_RECORDING_VOLUME	Error while setting the recording volume

## 9.4.2 Common

### HideSplashScreen

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The FB controls whether the splash screen is shown. By default, the splash screen is shown after the device has booted. The splash screen is set with the LoadSplashScreen function block.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

- Only call up one instance of the function block within the application!

### Input parameter

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
xHide	BOOL	Show/hide splash screen.	FALSE	Hide splash screen and show visualisation.
			TRUE	Show splash screen and hide visualisation.

### Output parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
stSettings	stAUDIO_SETTINGS	Volume settings of the device's audio channels.	→ stAUDIO_SETTINGS (STRUCT)	

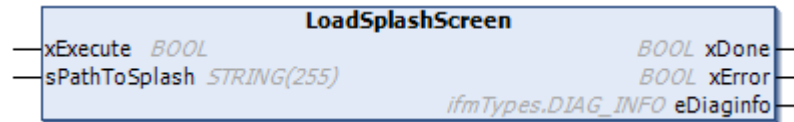
---

#### Diagnostic codes:

- |                          |   |
|--------------------------|---|
| • STAT_INACTIVE          | Status: FB/Function is inactive.  |
| • STAT_BUSY              | Status: FB/Function is currently executed.  |
| • STAT_DONE              | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs. |
| • ERR_INSTANCE_LIMIT     | Error: More than one instance of the FB has been created; this FB instance is not executed                |
| • ERR_HIDE_SPLASH_SCREEN | Error: Error while showing/hiding the splash screen.<br>Contact the ifm Service Center!                   |
| • ERR_UNDEFINED          | Error: Unknown error<br>Contact the ifm Service Center!   |

## LoadSplashScreen

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block loads an image file into the flash memory of the device. The image file will be used as splash screen the next time the device has rebooted.



Only one instance of the FB may be active within an application.

► Only call up one instance of the function block within the application!



Each call of the FB executes a write operation on the flash memory of the device.

► Do not call the function block cyclically in the program code!



Information on the image file for the splash screen:

- Image format: BMP 24 bit version 3.
- RLE compression and gzip compression are allowed.
- Recommended approach: Storage of the image file using MS Paint as 24 bit BMP.
- Recommended maximum image size: display resolution → Data sheet
- The file path is case-sensitive. Linux is case sensitive.
- The file name may only contain lower case letters.
- Transfer the new file to the unit using the CODESYS file browser or in setup mode.
- Maximum file size: unlimited. The image file uses a part of the available memory for the user application.

### Input parameter

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
sPathToSplash	STRING	Absolute directory path to the image file	e.g. '/home/cds-apps/PlcLogic/visu/splashscreen.bmp'	

### Output parameter

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed



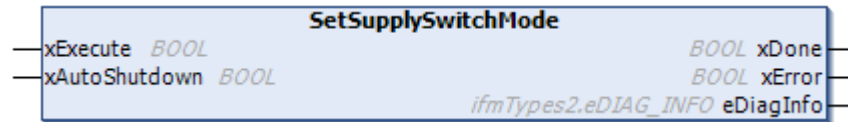
Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	TRUE	<ul style="list-style-type: none"> <li>• FB successfully executed</li> <li>• FB can be called again</li> </ul>
xError	BOOL	Indication if an error occurred during the FB execution	FALSE	No error occurred or the FB is still being executed
			TRUE	<ul style="list-style-type: none"> <li>• An error has 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                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT              Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_INVALID\_VALUE                Error: the file path for the splash screen on sPathToSplash is invalid. Function call has been stopped.  
Invalid values for sPathToSplash:
  - Value is ZERO (e.g. ")
  - Value at "/" (e.g. a directory)
  - Value has consecutive slashes (e.g. "///")
- ERR\_SET\_SPLASH\_SCREEN          Error: Error while loading the splash screen.  
Contact the ifm Service Center!
- ERR\_UNDEFINED                    Error: Unknown error  
Contact the ifm Service Center!

## SetSupplySwitchMode

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block enables or disables the automatic shut-down (AutoShutdown) of the device depending on the voltage value on terminal 15.

Default setting when the device has been switched on: Automatic shut-down is enabled. The device shuts down as soon as the voltage value on terminal 15 is < 50% of the voltage value on terminal 30.

If the automatic shut-down is disabled: The device remains switched on until the FB ShutdownDevice is called.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

- Only call up one instance of the function block within the application!

## Input parameter

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
xAutoShutdown	BOOL	Switch on / off automatic shut-down	FALSE	Automatic shut-down disabled
			TRUE.	Automatic shut-down enabled

## Output parameter

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>An error has 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      | Status: FB/Function is inactive.  |
| • STAT_BUSY          | Status: FB/Function is currently executed.  |
| • STAT_DONE          | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs. |
| • ERR_INSTANCE_LIMIT | Error: More than one instance of the FB has been created; this FB instance is not executed                |
| •                    | Error: error when enabling / disabling the automatic shut-down.<br>Contact the ifm Service Center!        |
| • ERR_UNDEFINED      | Error: Unknown error<br>Contact the ifm Service Center!   |



## ShutdownDevice

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block causes a controlled shut-down or reboot of the device depending on the voltage value on terminal 15.

Status /voltage value of terminal 15	Type of shot-down
OFF = FALSE => lower than 50% of the voltage value on terminal 30	Shut down the device
ON = TRUE / higher than approx. 4 V	Reboot the unit.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

- Only call up one instance of the function block within the application!

## Input parameter

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 parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

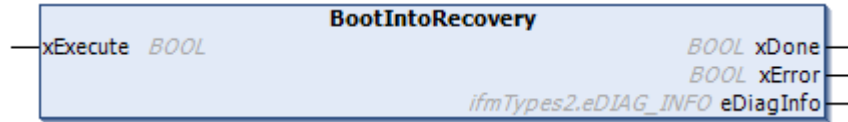
Diagnostic codes:

---

• STAT_INACTIVE	Status: FB/Function is inactive.
• STAT_BUSY	Status: FB/Function is currently executed.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_SHUTDOWN_DEVICE	Error: error during shut-down. Contact the ifm Service Center!
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## BootIntoRecovery

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block boots the device in the recovery mode. For this purpose, the device is immediately rebooted on command.



Data loss is possible: Data that is not stored (e.g. alarms, data logging, process data, recipes) will be lost when the device is rebooted.

- Store all important data on the FLASH memory of the device before the reboot.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

- Only call up one instance of the function block within the application!

## Input parameter

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 parameter

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>• An error has 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	Status: FB/Function is inactive.
• STAT_BUSY	Status: FB/Function is currently executed.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_SET_RECOVERY_MODE	Error: Error while setting the recovery mode. Contact the ifm Service Center!
• ERR_SHUTDOWN_DEVICE	Error: error during shut-down. Contact the ifm Service Center!
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## GetDeviceOrientation

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block provides the orientation (rotation) of the splash screen content as configured in the device.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

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 parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
eRotation	eOBJECT_ROTATION	Configured device orientation (rotation).	→ eOBJECT_ROTATION (ENUM)	

Diagnostic codes:

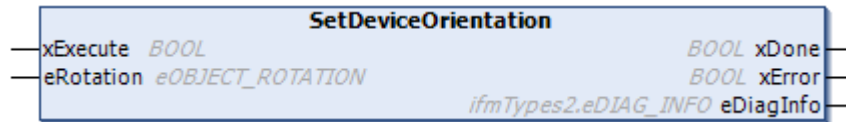
- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.



- 
- |                           |   |
|---------------------------|---|
| • STAT_DONE               | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs. |
| • ERR_INSTANCE_LIMIT      | Error: More than one instance of the FB has been created; this FB instance is not executed                |
| • ERR_GET_DEVICE_ROTATION | Error: Error while reading the configured device orientation.   |
| • ERR_UNDEFINED           | Error: Unknown error<br>Contact the ifm Service Center!   |

## SetDeviceOrientation

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The FB rotates the splash screen content depending on the device orientation. After successful execution of the function block and subsequent device reboot, the splash screen content is rotated according to the settings.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

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
eRotation	eOBJECT_ROTATION	Configured device orientation (rotation).	→ eOBJECT_ROTATION (ENUM)	

## Output parameter

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>• An error has 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                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.

- 
- |                           |   |
|---------------------------|---|
| • STAT_DONE               | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs. |
| • ERR_INSTANCE_LIMIT      | Error: More than one instance of the FB has been created; this FB instance is not executed                |
| • ERR_GET_DEVICE_ROTATION | Error: Error while reading the configured device orientation.   |
| • ERR_UNDEFINED           | Error: Unknown error<br>Contact the ifm Service Center!   |

### 9.4.3 Ethernet

#### GetEthernetInterfaces

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



#### Description

The function block provides a list with all available Ethernet interfaces of the device.

#### Input parameter

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 parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
usiEthItfsCnt	USINT	Number of available Ethernet interfaces	0	no Ethernet interface
			...	...
			15	15 Ethernet interfaces
aEthItfsList	aETH_ITF_LIST	List of the available Ethernet interfaces	→ aETH_ITF_LIST (GVL)	

Diagnostic codes:

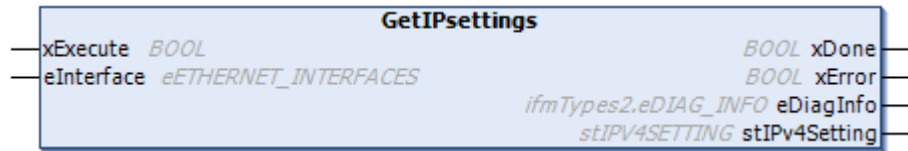
- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                            Status: FB/Function is currently executed.

---

• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_GET_ETH_ITF_LIST	Error: Problem when reading the available Ethernet interfaces
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## GetIPsettings

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block reads the following IP settings of the indicated Ethernet interface of the device:

- IP address
- Network mask of the TCP/IP network as well as
- IP address of the gateway
- DHCP status

The FB provides the read values in a complex variable of the "stIPv4Setting" type.

## Input parameter

Parameter	Data type	Meaning	Possible values
xExecute	BOOL	Control execution of the FB	<ul style="list-style-type: none"> <li>• FALSE =&gt; TRUE: FB is executed once</li> <li>• Otherwise: No impact on FB processing</li> </ul>
eInterface	eETHERNET_INTERFACES	Ethernet interface ID	eETHERNET_INTERFACES (ENUM) (→ □ 205)

## Output parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
stIPv4Setting	stIPv4Setting	IPv4 settings of the device	stIPv4SETTING (STRUCT) (→ □ 211)	

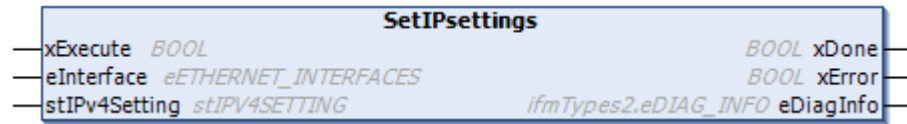
Diagnostic codes:

---

• STAT_INACTIVE	Status: FB/Function is inactive.
• STAT_BUSY	Status: FB/Function is currently executed.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INVALID_VALUE	Error: invalid Ethernet interface selected.
• ERR_GET_DHCP_STATUS	Error while getting the DHCP server status.
• ERR_GET_IP_SETTINGS	Error while getting the IP configuration.
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## SetIPsettings

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block changes the following IP settings of the indicated Ethernet interface of the device:

- IP address
- Subnet mask of the TCP/IP network
- IP address of the gateway
- DHCP status

The IP settings are transferred to the FB in a complex variable of the "stIPv4Setting" type.

The parameter DHCP has the highest priority. If DHCP = TRUE the values of the parameters IP address, subnet mask and gateway address are not evaluated.

After the call, the FB checks if DHCP is activated. If yes, the FB deactivates the DHCP client of the device and sets the required IP address.



- ▶ Enter the IP address using the following notation www.xxx.yyy.zzz
- ▶ Leave out the leading zeros when entering the IP address!  
FALSE: 192.168.000.055  
CORRECT: 192.168.0.55



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

- ▶ Only call up one instance of the function block within the application!



Each call of the FB executes a write operation on the flash memory of the device.

- ▶ Do not call the function block cyclically in the program code!

## Input parameter

Parameter	Data type	Meaning	Possible values
xExecute	BOOL	Control execution of the FB	<ul style="list-style-type: none"><li>• FALSE =&gt; TRUE: FB is executed once</li><li>• Otherwise: No impact on FB processing</li></ul>
eInterface	eETHERNET_INTERFACES	Ethernet interface ID	eETHERNET_INTERFACES (ENUM) (→ 205)
stIPv4Setting	stIPv4Setting	IPv4 settings of the device	stIPv4SETTING (STRUCT) (→ 211)



## Output parameter

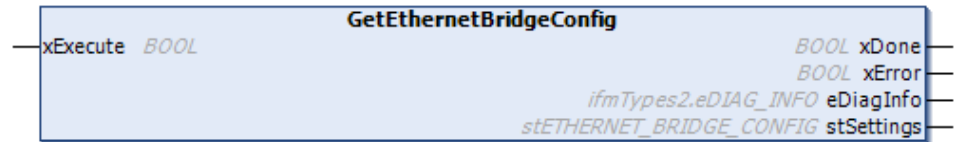
Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed
			TRUE	<ul style="list-style-type: none"> <li>• FB successfully executed</li> <li>• FB can be called again</li> </ul>
xError	BOOL	Indication if an 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>• An error has 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                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE                Error: invalid Ethernet interface selected.
- ERR\_GET\_DHCP\_STATUS            Error while getting the DHCP server status.
- ERR\_SET\_DHCP\_STATUS            Error while changing the DHCP server status.
- ERR\_SET\_IP\_SETTINGS             Error while writing the IP configuration.  
Check the IP address, subnet mask and default gateway.
- ERR\_UNDEFINED                    Error: Unknown error  
Contact the ifm Service Center!

## GetEthernetBridgeConfig

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block provides the Ethernet bridge mode configuration that is configured in the device.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

Parameter	Data type	Meaning	Possible values
xExecute	BOOL	Control execution of the FB	<ul style="list-style-type: none"><li>FALSE =&gt; TRUE: FB is executed once</li><li>Otherwise: No impact on FB processing</li></ul>

### Output parameter

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>An error has occurred</li><li>Action could not be executed</li><li>Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
stSettings	stETHERNET_BRIDGE_CONFIG	Ethernet bridge node configuration	→ stETHERNET_BRIDGE_CONFIG (STRUCT)	

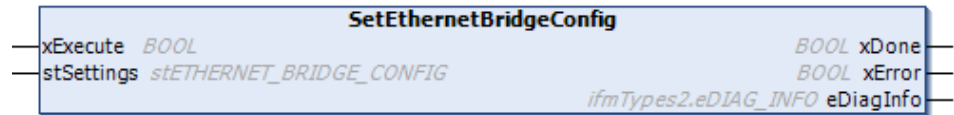
### Diagnostic codes:

- |                      |   |
|----------------------|---|
| • STAT_INACTIVE      | Status: FB/Function is inactive.  |
| • STAT_BUSY          | Status: FB/Function is currently executed.  |
| • STAT_DONE          | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs. |
| • ERR_INSTANCE_LIMIT | Error: More than one instance of the FB has been created; this FB instance is not executed                |

- 
- ERR\_BRIDGE\_MODE\_GET\_STAT  
US Error: Error while reading the Ethernet bridge mode status
  - ERR\_BRIDGE\_MODE\_NOT\_ACTIV  
E Error: Ethernet bridge mode is not enabled. The Ethernet bridge mode configuration cannot be read.
  - ERR\_GET\_DHCP\_STATUS Error: Error while reading the DHCP status.
  - ERR\_GET\_IP\_SETTINGS Error: Error while reading the current IP settings.
  - ERR\_UNDEFINED Error: Unknown error  
Contact the ifm Service Center!

## SetEthernetBridgeConfig

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The FB starts or stops the bridge mode between two networks connected to the Ethernet interfaces of the unit.

When the FB is executed, the Bridge Mode settings become active.

The following settings are possible for Bridge Mode:

- bridge network:
  - IP address
  - DHCP mode / static
- Selection of the participating Ethernet interfacea



► Enter the IP address using the following notation www.xxx.yyy.zzz

► Leave out the leading zeros when entering the IP address!

FALSE: 192.168.000.055

CORRECT: 192.168.0.55



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

Parameter	Data type	Meaning	Possible values
xExecute	BOOL	Control execution of the FB	<ul style="list-style-type: none"><li>• FALSE =&gt; TRUE: FB is executed once</li><li>• Otherwise: No impact on FB processing</li></ul>
stSettings	stETHERNET_BRIDGE_CONFIG	Ethernet bridge node configuration	stETHERNET_BRIDGE_CONFIG (STRUCT) (→ 210)

### Output parameter

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

Parameter	Data type	Description	Possible values
xError	BOOL	Indication if an error occurred during the FB execution	TRUE <ul style="list-style-type: none"> <li>An error has 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                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                          Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT              Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_BRIDGE\_MODE\_GET\_STAT US   Error: Error while reading the Ethernet bridge mode status
- ERR\_GET\_DHCP\_STATUS            Error: Error while reading the DHCP status.
- ERR\_SET\_DHCP\_STATUS            Error: Error while changing the DHCP status.
- ERR\_SET\_IP\_SETTINGS             Error: Error while changing the IP settings. Either the IP address, the subnet mask or the standard gateway is invalid.
- ERR\_BRIDGE\_MODE\_STOP           Error: Error while stopping the Ethernet bridge mode.
- ERR\_UNDEFINED                    Error: Unknown error  
Contact the ifm Service Center!

## 9.4.4 Keypads

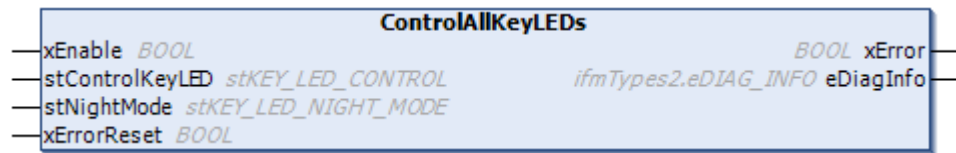
### ControlAllKeyLEDs

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmDevice\_ecomatDisplay.library

**Symbol in CODESYS:**



### Description

The function block switches on/off all key LEDs of the device in accordance with the selected settings and sets the LED colour.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
stNightMode	stKEY_LED_NIGHT_MODE	Enable/disable night mode for illumination of the key LEDs and set the night mode brightness.	→ stLED_SETTINGS (STRUCT)	
stControlKeyLED	stKEY_LED_CONTROL	Setting / controlling the key LEDs: On/off and colour	→ stKEY_LED_CONTROL (STRUCT)	
xErrorReset	BOOL	Reset the error if xError is enabled at the output function block.	FALSE => TRUE	Reset error
			FALSE	(Default value)

### Output parameter

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>An error has 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:

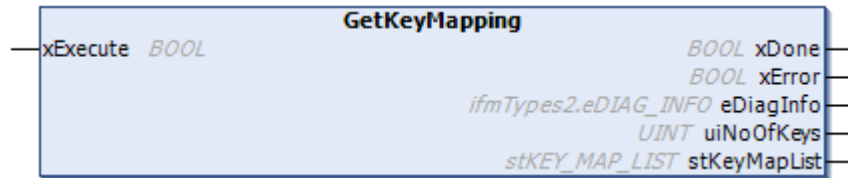
- STAT\_INACTIVE                      Status: FB/Function is inactive.

---

• STAT_PREPARING	Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_INVALID_VALUE	Error: access to the key LEDs not supported by the target device
• ERR_SET_KEY_LED_COLOR	Error while setting the LED colour. Contact the ifm Service Center!
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## GetKeyMapping

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block reads the current mapping settings of the integrated keypad.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

- Only call up one instance of the function block within the application!

### Input parameter

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 parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
uiNoOfKeys	UINT	Number of keys on the integrated keypad		
stKeyMapList	stKEY_MAP_LIST	Structure with mapping information of the keys.	→ stKEY_MAP_LIST (STRUCT)	

Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.

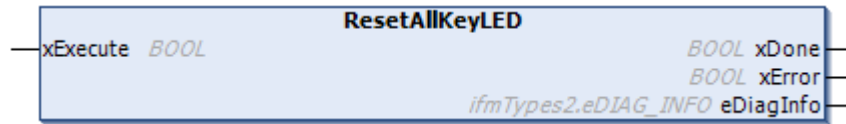


---

• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_GET_KEY_CONFIG	Error while reading the key configuration
• ERR_GET_KEY_MAP	Error while reading the key mapping
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## ResetAllKeyLED

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block resets all key LEDs to the default value "OFF".



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

- Only call up one instance of the function block within the application!

### Input parameter

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 parameter

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>An error has 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:

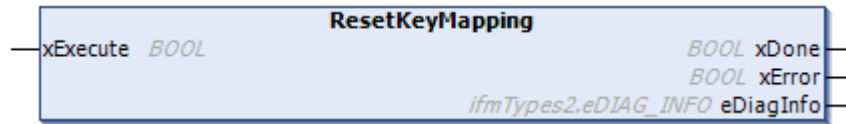
- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_BUSY      Status: FB/Function is currently executed.
- STAT\_DONE      Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT      Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_RESET\_KEY\_LED      Error: error when resetting the key LEDs.  
Contact the ifm Service Center!

- 
- ERR\_UNDEFINED

Error: Unknown error  
Contact the ifm Service Center!

## ResetKeyMapping

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block resets the mapping of all keys to the default values.



- ▶ Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.
- ▶ Only call up one instance of the function block within the application!

## Input parameter

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 parameter

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>An error has 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:

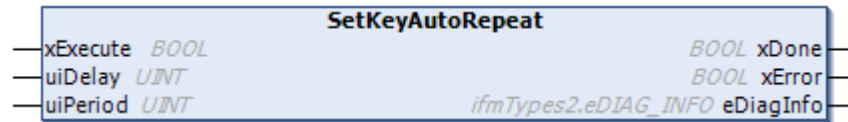
- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                          Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.

---

• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_RESET_KEY_MAP	<ul style="list-style-type: none"><li>• Error while resetting the key mapping</li></ul> Contact the ifm Service Center!
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## SetKeyAutoRepeat

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block sets the auto repeat configuration (pulse repetition) for the keys of the integrated keypad.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

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
uiDelay	UINT	Time in [ms] for which a key must be pressed until auto repeat will be started.		
uiPeriod	UINT	Time in [ms] between two auto repeat pulses.		

## Output parameter

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>• An error has 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:

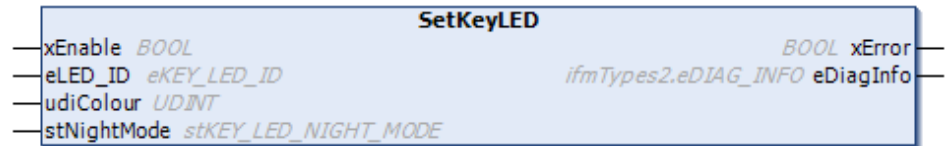
- STAT\_INACTIVE Status: FB/Function is inactive.

---

• STAT_BUSY	Status: FB/Function is currently executed.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_SET_KEY_AUTO_REPEAT	<ul style="list-style-type: none"> <li>• Error while setting the auto repeat configuration</li> </ul> Contact the ifm Service Center!
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## SetKeyLED

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block sets the colour and the night mode of the indicated key LEDs.

## Input parameter

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
eLED_ID	eKEY_LED_ID	Indication of the key LEDs on the integrated keypad.	→ eKEY_LED_ID (ENUM)	
udiColor	UDINT	LED colour to be set.	RGB: f 16#00RRGGBB with RR = 0...FF GG = 0...FF BB = 0...FF Examples: red = 16#00FF0000 green = 16#0000FF00 blue = 16#000000FF black = 16#00000000 white = 16#00FFFFFF	
stNightMode	stKEY_LED_NIGHT_MODE	Enable/disable night mode for illumination of the key LEDs and set the night mode brightness.	→ stLED_SETTINGS (STRUCT)	

## Output parameter

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>An error has 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 outputs are invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed



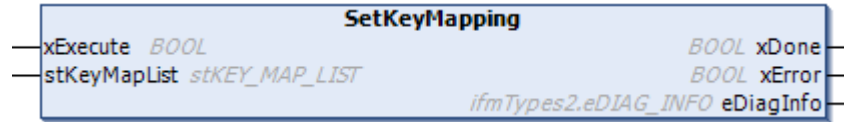
---

#### Diagnostic codes:

• STAT_INACTIVE	Status: FB/Function is inactive.
• STAT_BUSY	Status: FB/Function is currently executed.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INVALID_VALUE	Error: Invalid indication of the LED at the input parameter eLED_ID.
• ERR_SET_KEY_LED_COLOR	Error while setting the LED colour. Contact the ifm Service Center!
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## SetKeyMapping

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block assigns default keyboard functions to the keys of the integrated keypad (key mapping).



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

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
stKeyMapList	stKEY_MAP_LIST	Structure with mapping information of the keys.	→ stKEY_MAP_LIST (STRUCT)	

## Output parameter

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>• An error has 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:

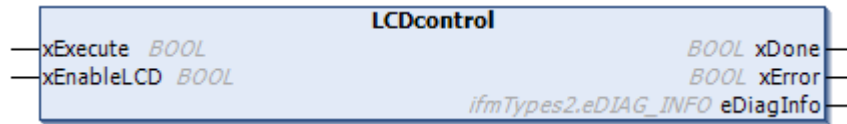
- |                 |   |
|-----------------|---|
| • STAT_INACTIVE | Status: FB/Function is inactive.  |
| • STAT_BUSY     | Status: FB/Function is currently executed.  |
| • STAT_DONE     | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs. |

- 
- |                      |  |
|----------------------|--|
| • ERR_INSTANCE_LIMIT | Error: More than one instance of the FB has been created; this FB instance is not executed |
| • ERR_SET_KEY_MAP    | Error while setting the key mapping  |
| • ERR_UNDEFINED      | Error: Unknown error<br>Contact the ifm Service Center!                                    |

## 9.4.5 LCD

### LCDcontrol

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block switches on/off the LCD display and the background illumination.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

- Only call up one instance of the function block within the application!

### Input parameter

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
xEnableLCD	BOOL	Switching on /off LCD (low power mode).	FALSE	Switch on LCD.
			TRUE	Switch off LCD.

### Output parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

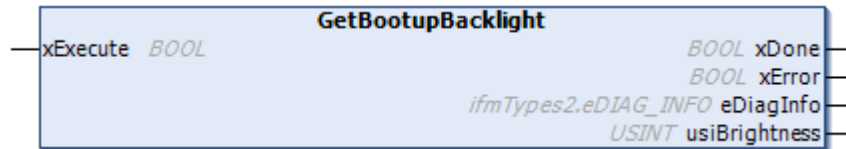
### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                          Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.

- 
- |                      |  |
|----------------------|--|
| • ERR_INSTANCE_LIMIT | Error: More than one instance of the FB has been created; this FB instance is not executed |
| • ERR_INTERNAL       | Error: Internal system error<br>Contact the ifm Service Center!                            |
| • ERR_UNDEFINED      | Error: Unknown error<br>Contact the ifm Service Center!                                    |

## GetBootupBacklight

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block reads the set LCD brightness settings for the boot process.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

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 parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
usiBrightness	USINT	Set brightness value for the device boot process	0...100 %	

### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                          Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.

- 
- |                                 |  |
|---------------------------------|--|
| • ERR_INSTANCE_LIMIT            | Error: More than one instance of the FB has been created; this FB instance is not executed |
| • ERR_GET_BOOTUP_LCD_BRIGHTNESS | Error while reading the brightness value<br>Contact the ifm Service Center!                |
| • ERR_UNDEFINED                 | Error: Unknown error<br>Contact the ifm Service Center!                                    |

## GetLCD\_Backlight

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block reads the set LCD brightness settings for normal display operation.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

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 parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
usiBrightness	USINT	Set brightness value for normal device operation	0...100 %	

### Diagnostic codes:

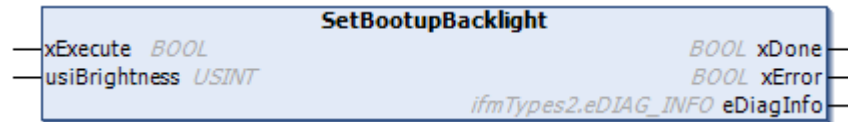
- |                      |   |
|----------------------|---|
| • STAT_INACTIVE      | Status: FB/Function is inactive.  |
| • STAT_BUSY          | Status: FB/Function is currently executed.  |
| • STAT_DONE          | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs. |
| • ERR_INSTANCE_LIMIT | Error: More than one instance of the FB has been created; this FB instance is not executed                |



- 
- `ERR_GET_LCD_BRIGHTNESS`      Error while reading the brightness value  
Contact the ifm Service Center!
  - `ERR_UNDEFINED`                Error: Unknown error  
Contact the ifm Service Center!

## SetBootupBacklight

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block sets the value for the LCD brightness for the boot process.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

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
usiBrightness	USINT	Brightness value for the device boot process	0...100 % Default value: 80 %	

### Output parameter

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>An error has occurred</li><li>Action could not be executed</li><li>Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

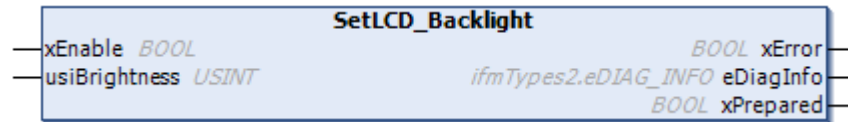
### Diagnostic codes:

- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_BUSY      Status: FB/Function is currently executed.
- STAT\_DONE      Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT      Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_INVALID\_VALUE      Error: value on the input parameter usiBrightness outside of the value range.

- 
- ERR\_SET\_BOOTUP\_LCD\_BRIGHTNESS  
Error while setting the brightness value.  
Contact the ifm Service Center!
  - ERR\_UNDEFINED  
Error: Unknown error  
Contact the ifm Service Center!

## SetLCD\_Backlight

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block sets the value for the LCD brightness for normal operation.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

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
usiBrightness	USINT	Brightness value for normal device operation	0...100 % Default value: 80 [%] Background illumination off = 0 [%]	

### Output parameter

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>An error has 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 outputs are invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed

#### Diagnostic codes:

- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_PREPARING      Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE      Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT      Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_INVALID\_VALUE      Error: value on the input parameter usiBrightness outside of the value range.

- 
- `ERR_SET_LCD_BRIGHTNESS`      Error while setting the brightness value.  
Contact the ifm Service Center!
  - `ERR_UNDEFINED`              Error: Unknown error  
Contact the ifm Service Center!

## 9.4.6 Local IO

### GetLightSensor

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block returns the light intensity value of the integrated light sensor.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

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 parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
uiAmbientLight	UINT	Value of the light intensity determined by the integrated light sensor	0...4096 = 0...100%	

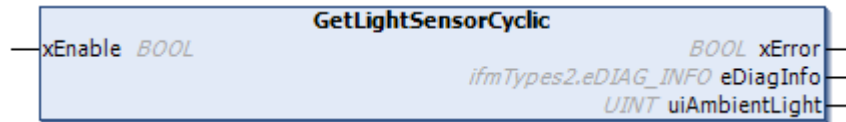
### Diagnostic codes:

- |                 |   |
|-----------------|---|
| • STAT_INACTIVE | Status: FB/Function is inactive.  |
| • STAT_BUSY     | Status: FB/Function is currently executed.  |
| • STAT_DONE     | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs. |

- 
- |                         |   |
|-------------------------|---|
| • ERR_INSTANCE_LIMIT    | Error: More than one instance of the FB has been created; this FB instance is not executed                                    |
| • ERR_READ_LIGHT_SENSOR | <ul style="list-style-type: none"><li>• Error while reading the light sensor value.</li></ul> Contact the ifm Service Center! |
| • ERR_UNDEFINED         | Error: Unknown error<br>Contact the ifm Service Center!   |

## GetLightSensorCyclic

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block returns the light intensity value of the integrated light sensor in a cyclical interval of 150 ms.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB

### Output parameter

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>An error has occurred</li><li>Action could not be executed</li><li>Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
uiAmbientLight	UINT	Value of the light intensity determined by the integrated light sensor	0...4096 = 0...100%	

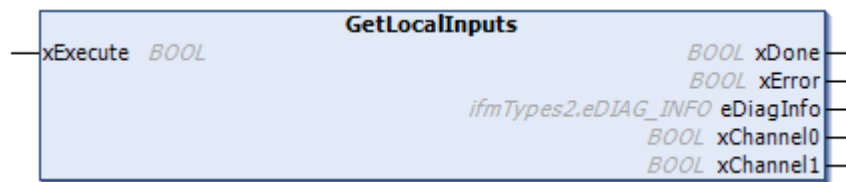
### Diagnostic codes:

- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_PREPARING      Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE      Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT      Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_READ\_LIGHT\_SENSOR
  - Error while reading the light sensor value.Contact the ifm Service Center!
- ERR\_UNDEFINED      Error: Unknown error  
Contact the ifm Service Center!



## GetLocalInputs

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block reads the state of the local digital inputs of the device.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

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 parameter

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>An error has occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xChannel0	BOOL	State of the digital input "Channel 0"	FALSE	OFF
			TRUE	ON
xChannel1	BOOL	State of the digital input "Channel 1"	FALSE	OFF
			TRUE	ON

## Diagnostic codes:

- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_BUSY      Status: FB/Function is currently executed.

- 
- |  |  |
|--|--|
| <ul style="list-style-type: none"><li>• STAT_DONE</li><li>• ERR_READ_DIGITAL_INPUT</li><li>• ERR_UNDEFINED</li></ul> | <p>Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.</p> <ul style="list-style-type: none"><li>• Error while reading the digital input channels.<br/>Contact the ifm Service Center!</li></ul> <p>Error: Unknown error<br/>Contact the ifm Service Center!</p> |
|--|--|

## GetLocalInputsCyclic

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



GB

### Description

The function block reads the state of the local digital inputs of the device at a cyclical interval of 50 ms.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB

### Output parameter

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>An error has occurred</li><li>Action could not be executed</li><li>Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xChannel0	BOOL	State of the digital input "Channel 0"	FALSE	OFF
			TRUE	ON
xChannel1	BOOL	State of the digital input "Channel 1"	FALSE	OFF
			TRUE	ON

### Diagnostic codes:

- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_PREPARING      Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE      Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_READ\_DIGITAL\_INPUT
  - Error while reading the digital input channels. Contact the ifm Service Center!

- 
- ERR\_UNDEFINED

Error: Unknown error  
Contact the ifm Service Center!

## GetTemperature

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block reads the following system temperatures:

- PCB temperature
- Processor temperature



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

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 parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
stTemperatures	stSYSTEM_TEMPERATURES	Structure with the system temperatures	→ stSYSTEM_TEMPERATURES (STRUCT)	

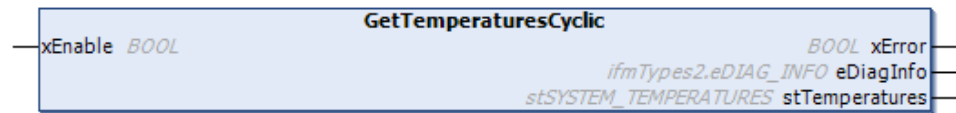
Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                            Status: FB/Function is currently executed.

- 
- STAT\_DONE                      Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
  - ERR\_INSTANCE\_LIMIT           Error: More than one instance of the FB has been created; this FB instance is not executed
  - ERR\_READ\_TEMPERATURE\_CORE\_0      Error while reading the processor temperature  
Contact the ifm Service Center!
  - ERR\_READ\_TEMPERATURE\_BOARD      Error while reading the PCB temperature.  
Contact the ifm Service Center!
  - ERR\_UNDEFINED                Error: Unknown error  
Contact the ifm Service Center!

## GetTemperaturesCyclic

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



GB

### Description

The function block reads the following system temperatures cyclically at an interval of 2000 ms:

- PCB temperature
- Processor temperature



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB

### Output parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
stTemperatures	stSYSTEM_TEMPERATURES	Structure with the system temperatures	→ stSYSTEM_TEMPERATURES (STRUCT)	

### Diagnostic codes:

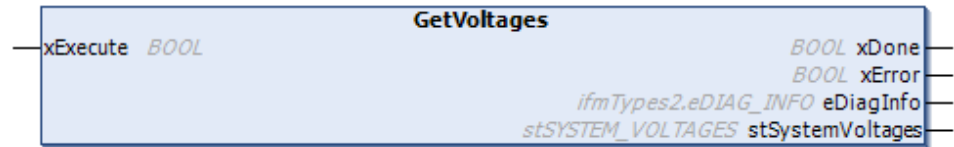
- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_PREPARING      Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE      Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT      Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_READ\_TEMPERATURE\_CORE\_0      Error while reading the processor temperature  
Contact the ifm Service Center!

- 
- ERR\_READ\_TEMPERATURE\_BOA RD Error while reading the PCB temperature.  
Contact the ifm Service Center!
  - ERR\_UNDEFINED Error: Unknown error  
Contact the ifm Service Center!



## GetVoltages

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



GB

## Description

The function block reads the following system voltages:

- VBB0
- VBB15
- VBB30



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

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 parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
stSystemVoltages	stSYSTEM_VOLTAGES	Structure with the system voltages	→ stSYSTEM_VOLTAGES (STRUCT)	

Diagnostic codes:

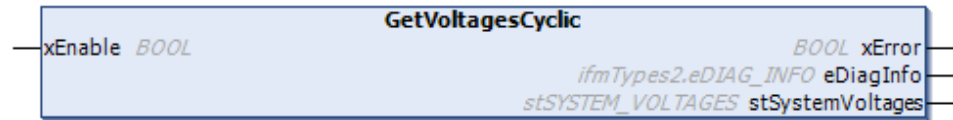
- STAT\_INACTIVE      Status: FB/Function is inactive.

---

• STAT_BUSY	Status: FB/Function is currently executed.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_READ_VOLTAGE_VBB0	Error while reading the system voltage VBB0. Contact the ifm Service Center!
• ERR_READ_VOLTAGE_VBB15	Error while reading the system voltage VBB15. Contact the ifm Service Center!
• ERR_READ_VOLTAGE_VBB30	Error while reading the system voltage VBB30. Contact the ifm Service Center!
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## GetVoltagesCyclic

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



GB

### Description

The function block reads the following system voltages cyclically at an interval of 50 ms:

- VBB0
- VBB15
- VBB30



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB

### Output parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
stSystemVoltages	stSYSTEM_VOLTAGES	Structure with the system voltages	→ stSYSTEM_VOLTAGES (STRUCT)	

### Diagnostic codes:

- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_PREPARING      Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE      Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT      Error: More than one instance of the FB has been created; this FB instance is not executed

- 
- ERR\_READ\_VOLTAGE\_VBB0      Error while reading the system voltage VBB0.  
Contact the ifm Service Center!
  - ERR\_READ\_VOLTAGE\_VBB15      Error while reading the system voltage VBB15.  
Contact the ifm Service Center!
  - ERR\_READ\_VOLTAGE\_VBB30      Error while reading the system voltage VBB30.  
Contact the ifm Service Center!
  - ERR\_UNDEFINED      Error: Unknown error  
Contact the ifm Service Center!

## SetLocalOutputs

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



GB

## Description

The function block writes the values of the local digital outputs. The function block has a diagnostic function. An error message is triggered and an error bit "xDiagChannel0 / 1" is set if the actual state of the outputs does not correspond with the target state.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
stChannel0	STRUCT	Command for digital output channel 0	→ stOUTPUT_COMMANDS (STRUCT)	
stChannel1	STRUCT	Command for digital output channel 1	→ stOUTPUT_COMMANDS (STRUCT)	

## Output parameter

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>An error has 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 outputs are invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
xErrorChannel0	BOOL	Error while writing on channel 0	FALSE	No error has occurred
			TRUE	An error has occurred
xDiagChannel0	BOOL	Diagnostic status of channel 0. Not relevant.	FALSE	Actual state = target state of channel 0
			TRUE	Actual state not equal to target state
xErrorChannel1	BOOL	Error while writing on channel 1	FALSE	No error has occurred

Parameter	Data type	Description	Possible values	
xErrorChannel1	BOOL	Error while writing on channel 1	TRUE	An error has occurred
xDiagChannel1	BOOL	Diagnostic status of channel 1. Not relevant.	FALSE	Actual state = target state of channel 1
			TRUE	Actual state not equal to target state

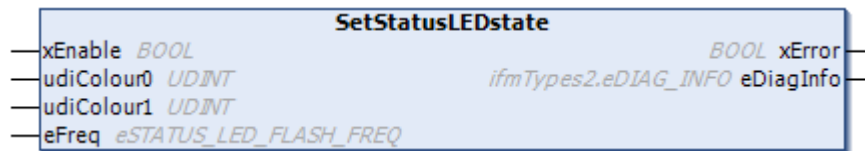
#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_PREPARING                    Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE                          Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT                Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_WRITE\_DIGITAL\_OUTPUT        Error while writing the digital output value.
- ERR\_READ\_DIAG\_INPUT              Error while reading the diagnostic status.
- ERR\_VALUE\_MISMATCH               Error: The set digital output value does not correspond with the read diagnostic state.
- ERR\_UNDEFINED                    Error: Unknown error  
Contact the ifm Service Center!

## 9.4.7 Status LED

### SetStatusLEDstate

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block controls the status LED of the device. The status LED can flash in two colours. The flashing frequency can be set.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

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
udiColor0	UDINT	First LED colour	RGB: f 16#00RRGGBB with RR = 0...FF GG = 0...FF BB = 0...FF Examples: red = 16#00FF0000 green = 16#0000FF00 blue = 16#000000FF black = 16#00000000 white = 16#00FFFFFF	
udiColor1	UDINT	Second LED colour	RGB: f 16#00RRGGBB with RR = 0...FF GG = 0...FF BB = 0...FF Examples: red = 16#00FF0000 green = 16#0000FF00 blue = 16#000000FF black = 16#00000000 white = 16#00FFFFFF	
eFreq	eSTATUS_LED_FLASH_FREQ	LED flashing frequency	→ eSTATUS_LED_FLASH_FREQ (ENUM)	

## Output parameter

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>• An error has 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                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT              Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_INVALID\_VALUE                Error: invalid value on udiColor0, udiColor1 or eFreq.
- ERR\_SET\_STATUS\_LED              Error while setting the LED colour.  
Contact the ifm Service Center!
- ERR\_UNDEFINED                    Error: Unknown error  
Contact the ifm Service Center!



## 9.4.8 Storage

### USBstorageHandler

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE with Busy Extension  
**Library:** ifmUSBstorageUtil.library  
**Symbol in CODESYS:**



### Description

The FB manages the USB device connected to the device. The FB carries out the following functions:

- Integrate USB device automatically into the file system of the device (mount)
- Provide path to the USB device in the file system of the device
- Remove USB device from the file system of the device upon command of the user (unmount)
- Signal insertion and removal of the USB device

### Input parameter

Parameter	Data type	Meaning	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
xRemoveDevice	BOOL	Remove USB device from the file system (unmount)	FALSE => TRUE	USB is removed
			Other	no action

### Output parameter

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>• An error has 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 outputs are invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
usiNumberUSBdevice	USINT	Number of the connected USB memory devices	0	no device connected
			1	1 device connected

Parameter	Data type	Meaning	Possible values	
sAccessPath	STRING	Absolute directory path of the USB storage device	e.g.. /mnt/usb	
xDevicePlugged	BOOL	Signals the insertion of a USB device within the current PLC cycle	FALSE	No USB device inserted
			TRUE	USB device inserted
xDeviceUnplugged	BOOL	Signals the removal of a USB device within the current PLC cycle	FALSE	USB device not removed
			TRUE	USB device has been removed

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_PREPARING                    Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE                            Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT                Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_INSTANCE\_RESTRICTION        Error: FB should not be used together with FB USBshandlerMulti.
- ERR\_INVALID\_VALUE                 Error: Wrong USB storage device was removed from file system
- ERR\_GET\_STORAGE\_LIST             Error: Problems while reading the list of connected USB memory devices
- ERR\_UNPLUG\_STORAGE              Error: Problems removing a USB storage device from the file system
- ERR\_UNDEFINED                      Error: Unknown error  
Contact the ifm Service Center!
- ERR\_INTERNAL                        Error: Internal system error  
Contact the ifm Service Center!

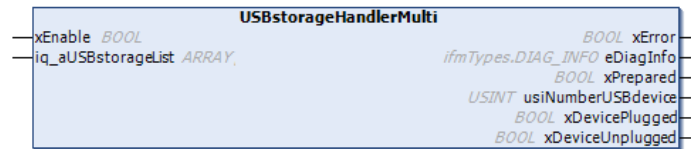
## USBstorageHandlerMulti

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE with Busy Extension

**Library:** ifmUSBstorageUtil.library

**Symbol in CODESYS:**



GB

### Description

The FB manages the USB devices connected to the device. The device-specific information of the USB devices is stored in an array. The FB carries out the following functions:

- Integrate USB devices automatically into the file system of the device (mount)
- Signal insertion and removal of the USB device
- Provide the paths to the USB devices in the file system of the device
- Remove USB devices from the file system of the device upon command of the user (unmount)

### Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
iq_aUSBstorageList	ARRAY [0..n] OF stUSB_STORAGE_INFO	Array with control signals and information about several USB devices	Per array row: → stUSB_STORAGE_INFO (STRUCT)	

### Output parameter

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>• An error has 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 outputs are invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
usiNumberUSBdevice	USINT	Number of the connected USB memory devices	0	no device connected
			1	1 device connected
xDevicePlugged	BOOL	Signals the insertion of a USB device within the current PLC cycle	FALSE	No USB device inserted

Parameter	Data type	Description	Possible values	
xDevicePlugged	BOOL	Signals the insertion of a USB device within the current PLC cycle	TRUE	USB device inserted
xDeviceUnplugged	BOOL	Signals the removal of a USB device within the current PLC cycle	FALSE	USB device not removed
			TRUE	USB device has been removed

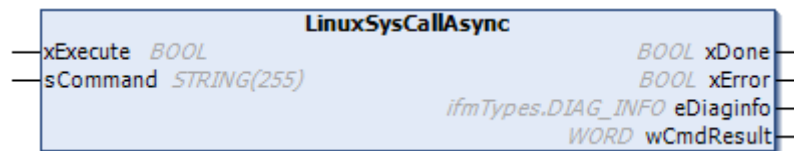
#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_PREPARING                  Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT              Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_INSTANCE\_RESTRICTION      Error: FB should not be used together with FB USBstorageHandler
- ERR\_INVALID\_VALUE               Error: Wrong USB storage device was removed from file system
- ERR\_GET\_STORAGE\_LIST           Error: Problems while reading the list of connected USB memory devices
- ERR\_UNPLUG\_STORAGE            Error: Problems removing a USB storage device from the file system
- ERR\_UNDEFINED                   Error: Unknown error  
Contact the ifm Service Center!
- ERR\_INTERNAL                     Error: Internal system error  
Contact the ifm Service Center!

## 9.4.9 System Commands

### LinuxSysCallAsync

Function block type: Function block (FB)  
 Behaviour model: EXECUTE  
 Library: ifmPDM360NGutil.library  
 Symbol in CODESYS:



### Description

The FB transmits a Linux command with parameters (max. 255 characters) to the operating system of the device and provides the return value of the command. The transmitted command is executed asynchronously in the background.

### Input parameter

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
sCommand	STRING	Linux command incl. parameters (max. 255 characters)		

### Output parameter

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>An error has occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
wCmdResult	WORD	Return value of the Linux command	Depends on the command, e.g.: 0: command has been executed successfully 1...65534: Error: execution of the command has failed. → CmpErrors.library or help for Linux command. 65535: Command is being executed	

Diagnostic code:

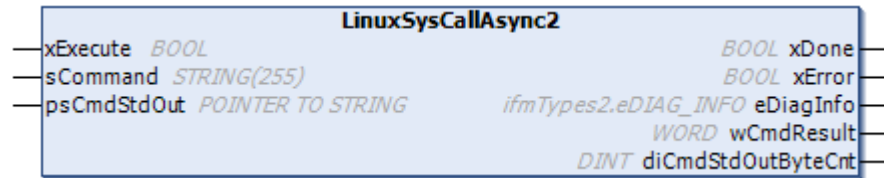
- STAT\_INACTIVE Status: FB/Function is inactive.

---

• STAT_BUSY	Status: FB/Function is currently executed.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INVALID_VALUE	Error: invalid value on sCommand input. Function call cancelled.
• ERR_LINUX_SYS_CALL	Error while executing the Linux command.
• ERR_ASYNC_TASK	Error: asynchronous task provides an error.

## LinuxSysCallAsync2

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The FB transmits a Linux command with parameters (max. 255 characters) to the operating system of the device and provides the return value of the command. The transmitted command is executed asynchronously in the background. The standard output (stdout) of the Linux command is stored in a string buffer.

### Properties of the buffer for the standard output

- possible buffer size 1...10000 bytes
- If the standard output of a Linux command is higher than the defined buffer size, not all information will be saved in the buffer and will partially be lost.
- Example with defined buffer size = 1000 bytes (→ following programming example):
  - In case of a standard output length of 1500 bytes, the bytes 0...499 of the standard output will be lost. The bytes 500...1499 of the standard output will then be contained in the buffer (psCmdStdOut).

### Programming example:

VAR

sCmdStdOut : STRING(1000); /// buffer size

instLinuxSysCallAsync2.psCmdStdOut : LinuxSysCallAsync2; /// instance of FB

END\_VAR;

instLinuxSysCallAsync2.psCmdStdOut := ADR(sCmdStdOut);

### Input parameter

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
sCommand	STRING	Linux command incl. parameters (max. 255 characters)		
psCmdStdOut	POINTER TO STRING	Address of the buffer for the standard output (stdout) of the Linux command. Possible size 1...10000 bytes.		

### Output parameter

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	FALSE	FB is executed

Parameter	Data type	Description	Possible values	
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
wCmdResult	WORD	Return value of the Linux command	Depends on the command, e.g.: 0: command has been executed successfully 1...65534: Error: execution of the command has failed. → CmpErrors.library or help for Linux command. 65535: Command is being executed	
diCmdStdOutputByteCnt	DINT	Length of the returned standard output in bytes.	0...10000 bytes	

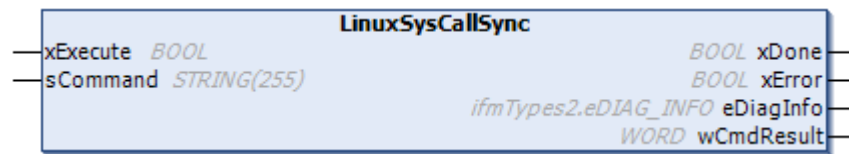
#### Diagnostic code:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                          Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE                Error: invalid value on sCommand input. Function call cancelled.
- ERR\_LINUX\_SYS\_CALL              Error while executing the Linux command.
- ERR\_ASYNC\_TASK                    Error: asynchronous task provides an error.



## LinuxSysCallSync

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



GB

### Description

The FB transmits a Linux command with parameters (max. 255 characters) to the operating system of the device and provides the return value of the command.

The transmitted command will be executed synchronously with the calling IEC task, this means:

- While the command is executed, the processing of the IEC task will be blocked.
  - Undesirable effects during the processing of the IEC application may be caused by a Linux command with a long duration, e.g. when copying large data volumes.
- Use the function blocks for asynchronous command execution in case of commands with a long duration:
- LinuxSysCallAsync (→ [165](#))
  - LinuxSysCallAsync2 (→ [167](#))

### Input parameter

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
sCommand	STRING	Linux command incl. parameters (max. 255 characters)		

### Output parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

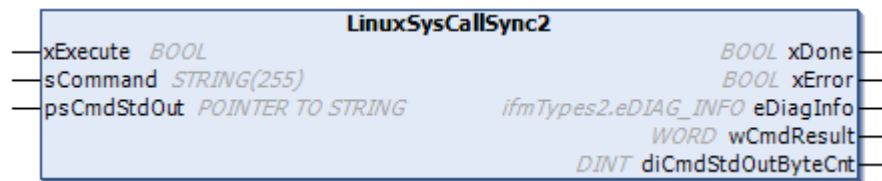
Parameter	Data type	Description	Possible values
wCmdResult	WORD	Return value of the Linux command	Depends on the command, e.g.: 0: command has been executed successfully 1...65534: Error: execution of the command has failed. → CmpErrors.library or help for Linux command. 65535: Command is being executed

#### Diagnostic code:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE                Error: invalid value on sCommand input. Function call cancelled.
- ERR\_LINUX\_SYS\_CALL              Error while executing the Linux command.

## LinuxSysCallSync2

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The FB transmits a Linux command with parameters (max. 255 characters) to the operating system of the device and provides the return value of the command. The standard output (stdout) of the Linux command is stored in a string buffer.

The transmitted command will be executed synchronously with the calling IEC task, this means:

- While the command is executed, the processing of the IEC task will be blocked.
  - Undesirable effects during the processing of the IEC application may be caused by a Linux command with a long duration, e.g. when copying large data volumes.
- Use the function blocks for asynchronous command execution in case of commands with a long duration:
- LinuxSysCallAsync (→ 165)
  - LinuxSysCallAsync2 (→ 167)

### Properties of the buffer for the standard output

- possible buffer size 1...10000 bytes
- If the standard output of a Linux command is higher than the defined buffer size, not all information will be saved in the buffer and will partially be lost.
- Example with defined buffer size = 1000 bytes (→ following programming example):
  - In case of a standard output length of 1500 bytes, the bytes 0...499 of the standard output will be lost. The bytes 500...1499 of the standard output will then be contained in the buffer (psCmdStdOut).

### Programming example:

```
VAR
sCmdStdOut : STRING(1000); /// buffer size
instLinuxSysCallSync2.psCmdStdOut : LinuxSysCallSync2; /// instance of FB
END_VAR;
instLinuxSysCallSync2.psCmdStdOut := ADR(sCmdStdOut);
```

### Input parameter

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

Parameter	Data type	Description	Possible values
sCommand	STRING	Linux command incl. parameters (max. 255 characters)	
psCmdStdOut	POINTER TO STRING	Address of the buffer for the standard output (stdout) of the Linux command. Possible size 1...10000 bytes.	

## Output parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
wCmdResult	WORD	Return value of the Linux command	Depends on the command, e.g.: 0: command has been executed successfully 1...65534: Error: execution of the command has failed. → CmpErrors.library or help for Linux command. 65535: Command is being executed	
diCmdStdOutputByteCnt	DINT	Length of the returned standard output in bytes.	0...10000 bytes	

## Diagnostic code:

- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_BUSY      Status: FB/Function is currently executed.
- STAT\_DONE      Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE      Error: invalid value on sCommand input. Function call cancelled.
- ERR\_LINUX\_SYS\_CALL      Error while executing the Linux command.

## 9.4.10 System Information

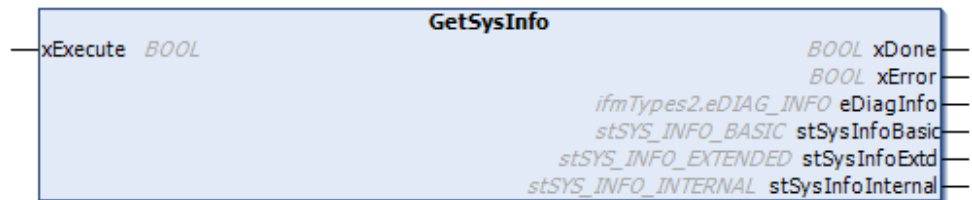
### GetSysInfo

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmDevice\_ecomatDisplay.library

**Symbol in  
CODESYS:**



### Description

The function block reads the system information from the device.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

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 parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
stSysInfoBasic	stSYS_INFO_BASIC	Basic system information that has been read	→ stSYS_INFO_BASIC (STRUCT)	
stSysInfoExtd	stSYS_INFO_EXTENDED	Extended system information that has been read	→ stSYS_INFO_EXTENDED (STRUCT)	
stSysInfoInternal	stSYS_INFO_INTERNAL	Internal system information (optional) that has been read.	→ stSYS_INFO_INTERNAL (STRUCT)	

Diagnostic codes:

---

• STAT_INACTIVE	Status: FB/Function is inactive.
• STAT_BUSY	Status: FB/Function is currently executed.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_GET_SYS_INFO	Error while reading the system information
• ERR_GET_KEYPAD_INFO	Error while reading the information about the keypad
• ERR_GET_KEY_INFO	Error while reading the key configuration
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## 9.4.11 Touch

### DisableTouchScreen

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The FB activates/deactivates the touch screen functionality of the display. The touch screen is enabled by default when the device will be rebooted.



The function block is only available for devices with an integrated touch screen.

### Input parameter

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
xDisableTouch	BOOL	Activate/deactivate the touch screen functionality of the display	FALSE	Touch screen functionality activated
			TRUE	Touch screen functionality deactivated

### Output parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

---

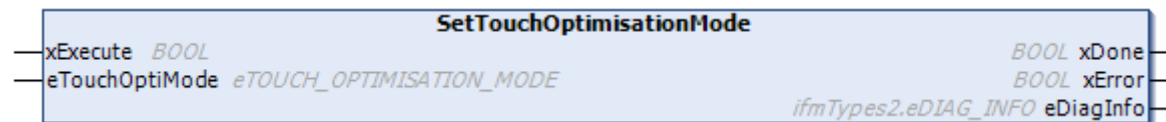
Diagnostic codes:

- |                      |   |
|----------------------|---|
| • STAT_INACTIVE      | Status: FB/Function is inactive.  |
| • STAT_BUSY          | Status: FB/Function is currently executed.  |
| • STAT_DONE          | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs. |
| • ERR_INSTANCE_LIMIT | Error: More than one instance of the FB has been created; this FB instance is not executed                |
| • ERR_INTERNAL       | Error: Internal system error<br>Contact the ifm Service Center!   |
| • ERR_UNDEFINED      | Error: Unknown error<br>Contact the ifm Service Center!   |



## SetTouchOptimisationMode

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block enables the optimisation mode for a certain operating condition. The user can choose between the following operating conditions:

- Standard operation
- Operation with gloves
- Operation in rain / splash water



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!



The function block is only available for devices with an integrated touch screen.

### Input parameter

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
eTouchOpti Mode	eTOUCH_OPTIMISATION_MODE	Optimisation mode for touch screen operation	→ eTOUCH_OPTIMISATION_MODE (ENUM)	

### Output parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

---

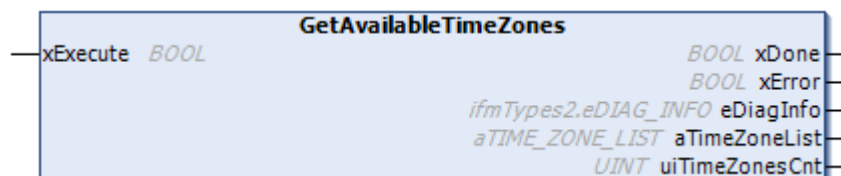
#### Diagnostic codes:

• STAT_INACTIVE	Status: FB/Function is inactive.
• STAT_BUSY	Status: FB/Function is currently executed.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_INVALID_VALUE	Error: Invalid optimisation mode selected
• ERR_INTERNAL	Error: Internal system error Contact the ifm Service Center!
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## 9.4.12 System Time

### GetAvailableTimeZones

Function block type: Function block (FB)  
 Behaviour model: EXECUTE  
 Library: ifmDevice\_ecomatDisplay.library  
 Symbol in CODESYS:



### Description

The FB provides the number and a list of the available time zone configurations of the unit.

### Input parameter

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 parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
aTimeZoneList	aTIME_ZONE_LIST	List of all available RTC time zones	→ aTIME_ZONE_LIST (ALIAS)	
uiTimeZonesCnt	UINT	Number of available RTC time zones	0	no RTC time zones
			...	...
			120	120 RTC time zones

Diagnostic codes:

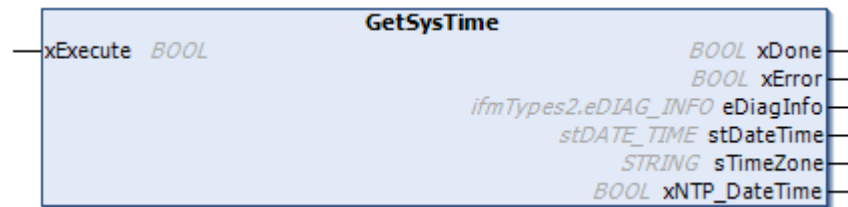
- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_BUSY      Status: FB/Function is currently executed.

---

• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_GET_TIMEZONES	Error: Problems while reading the time zone information: Inform the manufacturer
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## GetSysTime

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block reads the following parameters of the device-internal system clock and provides the values:

- Date
- Time
- Time zone
- NTP status



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

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 parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

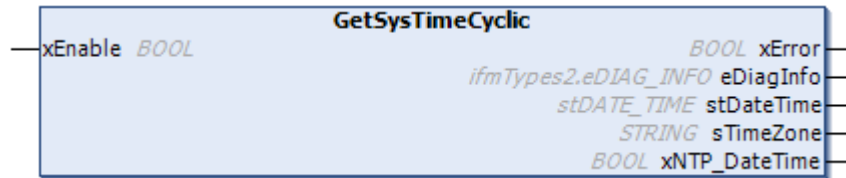
Parameter	Data type	Description	Possible values	
stDateTime	stDATE_TIME	Date and time	→ stDATE_TIME (STRUCT)	
sTimeZone	STRING	Time zone information		
xNTP_DateTime	BOOL	Update mode of date and time.	FALSE	Update via NTP not enabled.
			TRUE	Update via NTP is enabled.

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT              Error: More than one instance of the FB has been created; this FB instance is not executed
- ERR\_GET\_DATE\_TIME               Error: Error while reading date and/or time
- ERR\_GET\_TIMEZONES               Error: Error while reading the time zone information
- ERR\_UNDEFINED                    Error: Unknown error  
Contact the ifm Service Center!

## GetSysTimeCyclic

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



GB

### Description

The function block reads cyclically at an interval of 1000 ms the following parameters of the device-internal system clock and provides the values:

- Date
- Time
- Time zone
- NTP status



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

### Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB

### Output parameter

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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
stDateTime	stDATE_TIME	Date and time	→ stDATE_TIME (STRUCT)	
sTimeZone	STRING	Time zone information		
xNTP_DateTime	BOOL	Update mode of date and time.	FALSE	Update via NTP not enabled.
			TRUE	Update via NTP is enabled.

---

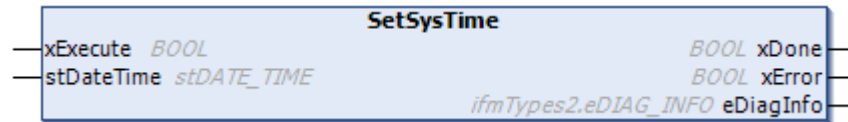
Diagnostic codes:

• STAT_INACTIVE	Status: FB/Function is inactive.
• STAT_PREPARING	Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_GET_DATE_TIME	Error: Error while reading date and/or time
• ERR_GET_TIMEZONES	Error: Error while reading the time zone information
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!



## SetSysTime

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



GB

### Description

The function block writes the following parameters of the system clock of the device:

- Date
- Time

Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

Only call up one instance of the function block within the application!

### Input parameter

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
stDateTime	stDATE_TIME	Date and time	→ stDATE_TIME (STRUCT)	

### Output parameter

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>• An error has 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:

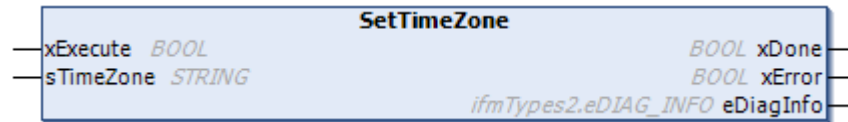
- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                              Status: FB/Function is currently executed.
- STAT\_DONE                              Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.

---

• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_SET_DATE_TIME	Error: Error while writing date and/or time
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## SetTimeZone

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



GB

### Description

The function block writes the following values of the system clock of the device:

- Time zone

The function block enables / disables the daylight saving time (DST) in accordance with the set time zone. If the automatic activation / deactivation of daylight saving time is not required, then select a time zone without daylight saving time (e.g. GMT+2).



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

- ▶ Only call up one instance of the function block within the application!

### Input parameter

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
sTimeZone	STRING	Time zone information		

### Output parameter

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>• An error has 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                      Status: FB/Function is inactive.
- STAT\_BUSY                            Status: FB/Function is currently executed.

---

• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_SET_TIMEZONES	Error: Error while writing the time zone information
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## GetNTP\_Settings

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block reads the configured NTP server settings and the NTP server status.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

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 parameter

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>An error has occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
stSettings	→ stNTP_SETTINGS	NTP settings	→ stNTP_SETTINGS	

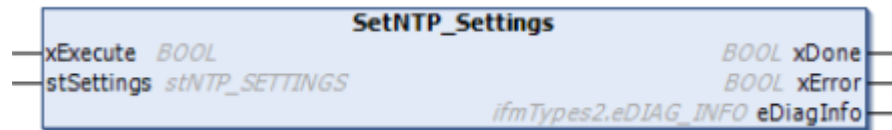
## Diagnostic codes:

- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_BUSY      Status: FB/Function is currently executed.
- STAT\_DONE      Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INSTANCE\_LIMIT      Error: More than one instance of the FB has been created; this FB instance is not executed

- 
- ERR\_GET\_NTP\_SERVERS                      Error: Error while reading the NRP servers configured in the system.
  - ERR\_GET\_NTP\_DAEMON\_STATUS              Error: Error while reading the NTP running status.
  - ERR\_UNDEFINED                              Error: Unknown error  
Contact the ifm Service Center!

## SetNTP\_Settings

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



GB

## Description

The function block enables / disables NTP services and configures the addresses of the NTP server in the system.



Only one instance of the FB may be active within an application. All function block calls will be cancelled, and an error message is given.

► Only call up one instance of the function block within the application!

## Input parameter

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
stSettings	→ stNTP_SETTINGS	NTP settings	→ stNTP_SETTINGS	

## Output parameter

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>• An error has 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 | Status: FB/Function is inactive.  |
| • STAT_BUSY     | Status: FB/Function is currently executed.  |
| • STAT_DONE     | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs. |

---

• ERR_INSTANCE_LIMIT	Error: More than one instance of the FB has been created; this FB instance is not executed
• ERR_INVALID_VALUE	Error: <ul style="list-style-type: none"> <li>• usiServerCnt &gt; 11</li> </ul> or <ul style="list-style-type: none"> <li>• aServerList[] = ZERO if usiServerCnt &gt; 0</li> </ul>
• ERR_SET_NTP_SERVERS	Error: Error while setting the NTP servers in the system
• ERR_GET_NTP_DAEMON_STATU S	Error: Error while reading the NTP-Daemon status from the system
• ERR_START_NTP_DAEMON	Error: Error while starting the NTP-Daemon
• ERR_STOP_NTP_DAEMON	Error: Error while stopping the NTP-Daemon
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!



## 9.4.13 Window Control

### AnalogueCameraWindowControl

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



### Description

The function block offers the following functions to configure and control an analogue camera:

- Start and stop
- Streaming the video image in the camera window
- Setting a Region of Interest (ROI) to be displayed in the camera window
- Rotating and mirroring the camera image
- Determining the size of the camera image
- Setting the background colour of the camera window to transparent

### Input parameter

Parameter	Data type	Description	Possible values
xEnable	BOOL	Control activity of the FB	FALSE   Deactivate FB
			TRUE   Enable FB
stWindowControls	STRUCT	Window configuration and control	→ stWINDOW_CONTROLS (STRUCT)
eCamera	eANALOGUE_VIDEO_STREAMS	Selection of the analogue camera video stream	→ eANALOGUE_VIDEO_STREAMS (ENUM)
stCameraControls	STRUCT	Configuration and control of the camera	→ stCAMERA_CONTROLS (STRUCT) (STRUCT)

### Output parameter

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>• An error has 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 outputs are invalid; FB is still processed
			TRUE   FB outputs valid; FB has been processed
stImageSize	STRUCT	Size of the camera image (determined)	→ stCAMERA_IMAGE_SIZE (STRUCT)

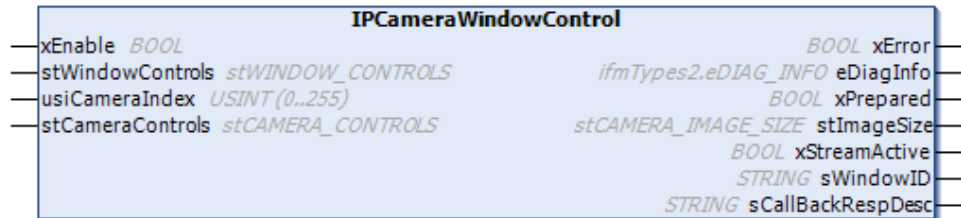
---

## Diagnostic codes:

• STAT_INACTIVE	Status: FB/Function is inactive.
• STAT_PREPARING	Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INVALID_VALUE	Error: access to the key LEDs not supported by the target device
• ERR_WM_SET_VISIBILITY	Error while setting the visibility of the window.
• ERR_WM_SET_POSITION_SIZE	Error while setting the position and size of the window.
• ERR_WM_BRING_TO_TOP	Error while setting the window to the foreground.
• ERR_WM_SEND_TO_BOTTOM	Error while setting the window to the background.
• ERR_WM_CAPTURE_WINDOW	Error while creating the screenshot.
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!
• ERR_CAMERA_START	Error while starting the camera.
• ERR_CAMERA_STOP	Error while stopping the camera.
• ERR_CAMERA_SET_FLIP_ROTATION	Error while setting the configuration of the camera image (rotating and mirroring)
• ERR_CAMERA_SET_ROI	Error while setting the ROI
• ERR_CAMERA_GET_IMAGE_SIZE	Error while reading the camera image size
• ERR_CAMERA_SET_BACKGROUND_COLOR	Error while setting the background colour

## IPCameraWindowControl

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block offers the following functions to configure and control an Ethernet camera:

- Start and stop
- Streaming the video image in the camera window
- Setting a Region of Interest (ROI) to be displayed in the camera window
- Rotating and mirroring the camera image
- Determining the size of the camera image
- Setting the background colour of the camera window to transparent

If `xEnable` = FALSE:

1. Stop camera.
2. After a set delay, output the set background colour in the camera window.
3. Hide camera window.

If `xEnable` = TRUE:

1. Start camera.
2. Display the camera stream in a camera window according to the parameterisation of the module.
3. Show camera window.

## Input parameter

Parameter	Data type	Description	Possible values	
sxEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
stWindowControls	STRUCT	Window configuration and control	→ stWINDOW_CONTROLS (STRUCT)	
usiCameraIndex	USINT	IP camera index. Each instance of the FB that is called up must be assigned a unique index value.	0...255	
stCameraControls	STRUCT	Configuration and control of the camera	→ stCAMERA_CONTROLS (STRUCT) (STRUCT)	

## Output parameter

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>An error has 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 outputs are invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed
stImageSize	STRUCT	Size of the camera image (determined)	→ stCAMERA_IMAGE_SIZE (STRUCT)	
xStreamActive	BOOL	Reception status of the camera video stream.	TRUE	Camera video stream is active and being received by the unit.
			FALSE	The camera video stream is interrupted. (This is not interpreted as an error.)
sWindowID	STRING	For experienced users: Window ID of the camera window	--	
sCallBackRespDesc	STRING	For experienced users: In case of IP camera fault - callback response description.	--	

## Diagnostic codes:



In the case of error messages "ERR\_" that occur during projection with frequent changes to the configuration and loading/restarting of the application:

- ▶ Disconnect the device and the camera from power and switch them on again after approx. 10 seconds.
- ▶ If the error persists and no cause can be identified: Contact the ifm Service Center!

• STAT_INACTIVE	Status: FB/Function is inactive.
• STAT_PREPARING	Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• STAT_EVNT_OPEN_CAMERA_ERROR	Status: The FB opens a Call Back Event for an IP camera error.
• STAT_EVNT_OPEN_CAMERA_STREAM_STATUS	Status: The FB opens a Call Back Event for changing the operating state of the IP camera stream.
• STAT_EVNT_OPEN_DONE	Status: The Call Back Event has been opened.
• STAT_EVNT_REGI_CAMERA_ERROR	Status: The FB registers a Call Back Function for an IP camera error.
• STAT_EVNT_REGI_CAMERA_STREAM_STATUS	Status: The FB registers a Call Back Function for changing the operating status of the IP camera stream.
• STAT_EVNT_REGI_DONE	Status: The all Back Function is registered.
• STAT_EVNT_CAMERA_STREAM_ACTIVATED	Status: The event "Activation of the IP camera stream" has occurred.
• STAT_EVNT_CAMERA_STREAM_INTERRUPTED	Status: The event "IP camera stream interruption" has occurred.

---

• ERR_INVALID_VALUE	Error: access to the key LEDs not supported by the target device
• ERR_WM_SET_VISIBILITY	Error while setting the visibility of the window.
• ERR_WM_SET_POSITION_SIZE	Error while setting the position and size of the window.
• ERR_WM_BRING_TO_TOP	Error while setting the window to the foreground.
• ERR_WM_SEND_TO_BOTTOM	Error while setting the window to the background.
• ERR_WM_CAPTURE_WINDOW	Error while creating the screenshot.
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!
• ERR_CAMERA_INIT_WINDOW	Error when initialising the camera after restarting the unit.
• ERR_CAMERA_START	Error while starting the camera.
• ERR_CAMERA_STOP	Error while stopping the camera.
• ERR_CAMERA_SET_FLIP_ROTATION	Error while setting the configuration of the camera image (rotating and mirroring)
• ERR_CAMERA_SET_ROI	Error while setting the ROI
• ERR_CAMERA_GET_IMAGE_SIZE	Error while reading the camera image size
• ERR_CAMERA_SET_BACKGROUND_COLOR	Error while setting the background colour
• ERR_EVT_OPEN	Error when opening an event for the Call Back Function registration.
• ERR_EVT_REGI	Error when registering a Call Back Function for a Call Back Event.

## PDF\_Viewer

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The FB provides functions to configure and display a PDF document in a PDF viewer window on the device, e.g.:

- Search
- Display a required page: First / last page, previous / next page, page above page number
- Scroll up or down
- Rotate left or right
- Zooming in and out, zooming to a specific value or zooming to the appropriate width
- Show / hide table of contents
- Full screen mode on / off

If the FB is enabled ( `xEnable = FALSE` ), the PDF and the window of the PDF viewer are closed.

If the name or path of the opened PDF ( `sFileName` ) is changed, the PDF is closed and the new PDF is opened.



In case of PDF errors (ERR\_PDF\_...):

- Disable the FB ( `xEnable = FALSE` ) to reset the error.

## Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
stWindowControls	STRUCT	Window configuration and control	→ stWINDOW_CONTROLS (STRUCT)	
sFileName	STRING	Name and path of the PDF file.	e.g. /home/example.pdf	
iq_stPdfControls	STRUCT	Configuration and control commands for the display of the PDF viewer.	→ stPDF_CONTROLS (STRUCT)	

## Output parameter

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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>

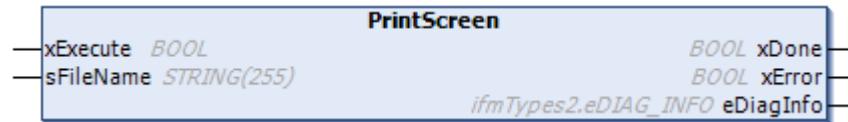
Parameter	Data type	Description	Possible values
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)
xPrepared	BOOL	State of the FB outputs	FALSE FB outputs are invalid; FB is still processed
			TRUE FB outputs valid; FB has been processed
udiWindowId	STRING	Fenster-ID des PDF-Viewer-Fensters	16#1003 0000... 16#1003 FFFF

## Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_PREPARING                    Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
- STAT\_DONE                            Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE                   Error: access to the key LEDs not supported by the target device
- ERR\_WM\_SET\_VISIBILITY              Error while setting the visibility of the window.
- ERR\_WM\_SET\_POSITION\_SIZE           Error while setting the position and size of the window.
- ERR\_WM\_BRING\_TO\_TOP                Error while setting the window to the foreground.
- ERR\_WM\_SEND\_TO\_BOTTOM             Error while setting the window to the background.
- ERR\_WM\_CAPTURE\_WINDOW             Error while creating the screenshot.
- ERR\_UNDEFINED                        Error: Unknown error  
Contact the ifm Service Center!
- ERR\_PDF\_VIEWER\_START                Error when starting or activating the PDF viewer.
- ERR\_PDF\_VIEWER\_STOP                Error when quitting of deactivating the PDF viewer.
- ERR\_PDF\_DOCU\_OPEN                  Error when opening a PDFdocument.
- ERR\_PDF\_DOCU\_CLOSE                 Error when closing a PDFdocument.
- ERR\_PDF\_GOTO\_PAGE                   Error during one of the following operation within a PDFdocument:
  - Displaying an indicated page
  - Displaying the first or the last page
  - Display of the previous or following page
- ERR\_PDF\_ROTATE\_PAGE                 Error when rotating the PDF document.
- ERR\_PDF\_SCROLL\_PAGE                 Error when scrolling the PDF document.
- ERR\_PDF\_SEARCH\_STRING               Error when searching for the entered string in the PDF document.
- ERR\_PDF\_SHOW\_HIDE\_TOC              Error when displaying / hiding the table of contents.
- ERR\_PDF\_ZOOM\_CONTROLS              Error when zooming then PDF document.
- ERR\_PDF\_SET\_FULL\_SCREEN\_MODE      Error when activating or deactivating full screen mode.

## PrintScreen

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block creates a screenshot of the display content and stores this as an image file.

Supported image formats:

- bmp (raw format = large file)
- jpg (preferred format for contents with camera image = small file)
- png (preferred format for HMI visualisation without camera images = small file)

## Input parameter

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
sFileName	STRING (255)	Directory path, name and format of the image file.	e.g. '/home/cds-apps/screenshot.bmp'	

## Output parameter

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>• An error has 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:

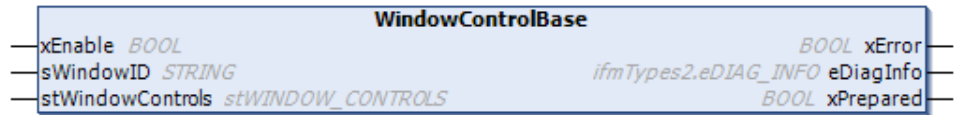
- |                 |   |
|-----------------|---|
| • STAT_INACTIVE | Status: FB/Function is inactive.  |
| • STAT_BUSY     | Status: FB/Function is currently executed.  |
| • STAT_DONE     | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs. |



- 
- |   |   |
|---|---|
| <ul style="list-style-type: none"><li>• ERR_INVALID_VALUE</li></ul>     | <p>Error: Value on input parameter sFileName is invalid. Function call has been stopped.</p> <p>Invalid values:</p> <ul style="list-style-type: none"><li>• The value at sFileName is ZERO (e.g. ")</li><li>• The value at sFileName ends with "/" (e.g. adirectory)</li><li>• The value on sFileName contains consecutive slashes (e.g. "////")</li><li>• The directory or subdirectory does not exist</li></ul> |
| <ul style="list-style-type: none"><li>• ERR_WM_CAPTURE_SCREEN</li></ul> | <p>Error while creating the screenshot.</p>   |
| <ul style="list-style-type: none"><li>• ERR_UNDEFINED</li></ul>         | <p>Error: Unknown error</p> <p>Contact the ifm Service Center!</p>  |

## WindowControlBase

**Function block type:** Function block (FB)  
**Behaviour model:** ENABLE  
**Library:** ifmDevice\_ecomatDisplay.library  
**Symbol in CODESYS:**



## Description

The function block offers the following functions to configure and control windows:

- Move to the foreground
- Move to the background
- Show or hide
- Set position and size
- Release and fix position
- Release and fix size
- Move a level further to the foreground
- Move a level further to the background
- Create a screenshot of the window content and save it as a file

## Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
sWindowID	STRING	Window ID or name		
stWindowControls	stWINDOW_CONTROLS	Window configuration and control	→ stWINDOW_CONTROLS (STRUCT)	

## Output parameter

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>• An error has 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 outputs are invalid; FB is still processed
			TRUE	FB outputs valid; FB has been processed

Diagnostic codes:

---

• STAT_INACTIVE	Status: FB/Function is inactive.
• STAT_PREPARING	Status: FB/FUN is processed; final results are not yet available. Some output values are updated in each PLC cycle.
• STAT_DONE	Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
• ERR_INVALID_VALUE	Error: access to the key LEDs not supported by the target device
• ERR_WM_SET_VISIBILITY	Error while setting the visibility of the window.
• ERR_WM_SET_POSITION_SIZE	Error while setting the position and size of the window.
• ERR_WM_BRING_TO_TOP	Error while setting the window to the foreground.
• ERR_WM_SEND_TO_BOTTOM	Error while setting the window to the background.
• ERR_WM_CAPTURE_WINDOW	Error while creating the screenshot.
• ERR_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## 9.4.14 ENUM

### CAN\_BAUDRATE (ENUM)

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)

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

### eANALOGUE\_VIDEO\_STREAMS (ENUM)

Name	Description	Possible values		Data type	Value
eANALOGUE_VIDEO_STREAMS	List of available analogue video streams.	CAM_0	Video stream analogue camera 0	INT	0
		CAM_1	Video stream analogue camera 1	INT	1
		CAM_2	Video stream analogue camera 2	INT	2
		CAM_3	Video stream analogue camera 3	INT	3

### eDAY\_OF\_WEEK (ENUM)

Name	Description	Possible values	Data type	Value
eDAY_OF_WEEK	List with days of the week	SUNDAY	USINT	0
		MONDAY	USINT	1
		TUESDAY	USINT	2
		WEDNESDAY	USINT	3
		THURSDAY	USINT	4
		FRIDAY	USINT	5

Name	Description	Possible values	Data type	Value
eDAY_OF_WEEK	List with days of the week	SATURDAY	USINT	6

### eETHERNET\_INTERFACES (ENUM)

Name	Description	Possible values		Data type	Value
eETHERNET_INTERFACES	List of the available Ethernet interfaces	ETH_0	Ethernet internet 0 (eth0)	INT	0
		ETH_1	Ethernet internet 1 (eth1)	INT	1

### eIP\_CAMERA\_ENCODING (ENUM)

Name	Description	Possible values		Data type	Value
eIP_CAMERA_ENCODING	List of the available IP camera stream codings	NONE	no selection	INT	0
		JPEG	JPEG coding	INT	1
		H264	H264 coding	INT	2

### eIP\_CAMERA\_PROTOCOL (ENUM)

Name	Description	Possible values		Data type	Value
eCAMERA_PROTOCOL	List of the available IP camera protocols	NONE	no selection	INT	0
		RTSP	Real Time Streaming Protocol	INT	1
		RTP	Real Time Protocol	INT	2

### eKEY\_LED\_ID (ENUM)

Name	Description	Possible values		Data type	Value
eKEY_LED_ID	List of the key LED IDs of the integrated keypad	NONE	no selection	UDINT	16#FF
		KEY0_0	LED key 0, group 0	UDINT	16:0
		KEY0_1	LED key 1, group 0	UDINT	16:2
		KEY0_2	LED key 2, group 0	UDINT	16:4
		KEY0_3	LED key 3, group 0	UDINT	16:6
		KEY0_4	LED key 4, group 0	UDINT	16:8
		KEY1_0	LED key 0, group 1	UDINT	16:1
		KEY1_1	LED key 1, group 1	UDINT	16:3
		KEY1_2	LED key 2, group 1	UDINT	16:5
		KEY1_3	LED key 3, group 1	UDINT	16:7
		KEY1_4	LED key 4, group 1	UDINT	16:9
		KEY2_NAVIGATION	LED navigation keys	UDINT	16 B

### eKEY\_MAP\_CODES (ENUM)

Name	Description	Possible values		Data type	Linux input event code (decimal)
eKEY_MAP_CODES	List of the possible standard keyboard functions (MF2) that can be assigned to a key on the integrated keypad.	NONE	no selection	UDINT	0
		ESCAPE	Escape key	UDINT	1
		MINUS	-	UDINT	74
		ASTERISK	*	UDINT	55
		SPACE	Space bar	UDINT	57

Name	Description	Possible values		Data type	Linux input event code (decimal)
eKEY_MAP_CODES	List of the possible standard keyboard functions (MF2) that can be assigned to a key on the integrated keypad.	COMMA	,	UDINT	83
		SLASH	/	UDINT	98
		PLUS	+	UDINT	78
		KEY_0	0	UDINT	11
		KEY_1	1	UDINT	2
		KEY_2	2	UDINT	3
		KEY_3	3	UDINT	4
		KEY_4	4	UDINT	5
		KEY_5	5	UDINT	6
		KEY_6	6	UDINT	7
		KEY_7	7	UDINT	8
		KEY_8	8	UDINT	9
		KEY_9	9	UDINT	10
		KEY_A	A	UDINT	30
		KEY_B	B	UDINT	48
		KEY_C	C	UDINT	46
		KEY_D	D	UDINT	32
		KEY_E	E	UDINT	18
		KEY_F	F	UDINT	33
		KEY_G	G	UDINT	34
		KEY_H	H	UDINT	35
		KEY_I	I	UDINT	23
		KEY_J	J	UDINT	36
		KEY_K	K	UDINT	37
		KEY_L	L	UDINT	38
		KEY_M	M	UDINT	50
		KEY_N	N	UDINT	49
		KEY_O	O	UDINT	24
		KEY_P	P	UDINT	25
		KEY_Q	Q	UDINT	16
		KEY_R	R	UDINT	19
		KEY_S	S	UDINT	31
		KEY_T	T	UDINT	20
		KEY_U	U	UDINT	22
		KEY_V	V	UDINT	47
		KEY_W	W	UDINT	17
		KEY_X	X	UDINT	45
		KEY_Y	Y	UDINT	21
		KEY_Z	Z	UDINT	44
		KEY_RETURN	Return key	UDINT	28
		BACKSPACE	Backspace key	UDINT	14
		TAB	Tab key	UDINT	15
		PRINT	Print key	UDINT	99

Name	Description	Possible values		Data type	Linux input event code (decimal)
eKEY_MAP_CODES	List of the possible standard keyboard functions (MF2) that can be assigned to a key on the integrated keypad.	HOME	Home key	UDINT	102
		UP	Arrow up key	UDINT	103
		LEFT	Arrow left key	UDINT	105
		RIGHT	Arrow right key	UDINT	106
		END	End key	UDINT	107
		DOWN	Arrow down key	UDINT	108
		INSERT	Insert key	UDINT	110
		DELETE	Delete key	UDINT	111
		BREAK	Pause key	UDINT	119
		F1	F1 key	UDINT	59
		F2	F2 key	UDINT	60
		F3	F3 key	UDINT	61
		F4	F4 key	UDINT	62
		F5	F5 key	UDINT	63
		F6	F6 key	UDINT	64
		F7	F7 key	UDINT	65
		F8	F8 key	UDINT	66
		F9	F9 key	UDINT	67
		F10	F10 key	UDINT	68
		F11	F11 key	UDINT	87
		F12	F12 key	UDINT	88
		NUM0	0 number pad	UDINT	82
		NUM1	1 number pad	UDINT	79
		NUM2	2 number pad	UDINT	80
		NUM3	3 number pad	UDINT	81
		NUM4	4 number pad	UDINT	75
		NUM5	5 number pad	UDINT	76
		NUM6	6 number pad	UDINT	77
		NUM7	7 number pad	UDINT	71
		NUM8	8 number pad	UDINT	72
		NUM9	9 number pad	UDINT	73

### eOBJECT\_FLIP (ENUM)

Name	Description	Possible values		Data type	Value
eOBJECT_FLIP	List of available flip settings for analogue video streams.	NONE	No selection.	INT	0
		NORMAL	Normal visualisation of the camera image.	INT	1
		FLIP	Mirrored visualisation of the camera image.	INT	2

### eOBJECT\_ROTATION (ENUM)

Name	Description	Possible values		Data type	Value
eOBJECT_ROTATION	List of the available rotation angles.	NONE	No selection.	INT	0
		DEG_0	Original position (0 degrees, "12 o'clock")	INT	1
		DEG_90	Clockwise rotation by 90 degrees	INT	2
		DEG_180	Clockwise rotation by 180 degrees	INT	3
		DEG_270	Clockwise rotation by 270 degrees	INT	4

### eSTATUS\_LED\_FLASH\_FREQ (ENUM)

Name	Description	Possible values		Data type	Value
eSTATUS_LED_FLASH_FREQ	List of flashing frequencies of the status LEDs	FREQ_0Hz	not flashing	INT	0
		FREQ_05Hz	0.5 Hz	INT	1
		FREQ_1Hz	1 Hz	INT	2
		FREQ_2Hz	2 Hz	INT	3
		FREQ_5Hz	5 Hz	INT	4

### eTOUCH\_OPTIMISATION\_MODE (ENUM)

Name	Description	Possible values		Data type	Value
eTOUCH_OPTIMISATION_MODE	Optimisation modes for touch screen operation	NONE	No optimisation	INT	0
		GLOVE	Optimisation for operation with glove	INT	1
		WATER	Optimisation for operation in rain / splash water	INT	2

## 9.4.15 STRUCT

### stAUDIO\_CHANNEL\_VOLUMES

Designation	Data type	Meaning	Possible values
	USINT	Left loudspeaker volume	0...100 %, Start value = 10 %
usiVolumeRight	USINT	Right loudspeaker volume	0...100 %, Start value = 10 %

### stAUDIO\_SETTINGS (STRUCT)

Designation	Data type	Meaning	Possible values
stMaster	→ stAUDIO_CHANNEL_VOLUMES	Master volume	0...100 %, Start value = 10 %
stSpeaker		Loudspeaker volume	0...100 %, Start value = 10 %
stHeadPhone		Headphones volume	0...100 %, Start value = 10 %
stLineInToLineOut		Volume Aux LineIn to LineOut channel	0...100 %, Start value = 10 %
stRecording		Recording channel volume	0...100 %, Start value = 10 %



## stCAMERA\_CONFIG (STRUCT)

Parameter	Data type	Meaning	Possible values
stROI	stCAMERA_ROI_CONFIG	Region of Interest (ROI) configuration in the camera image.	→ stCAMERA_ROI_CONFIG (STRUCT)
eFlip	eOBJECT_FLIP	Mirror setting	→ eOBJECT_FLIP (ENUM)
eRotation	eOBJECT_ROTATION	Rotation setting	→ eOBJECT_ROTATION (ENUM)
udiBackgroundColour	UDINT	Window background colour if no camera image is displayed (camera stopped) in RGBA format.	16#RRGGBBAA with red RR = 0...FF green GG = 0...FF blue BB = 0...FF Transparency AA: 0...FF Examples: 16#FFFFFF = black 100 % full (no transparency) 16#FFFFFF7F = white with 50 % transparency 16#0000007F = black with 50 % transparency
eProtocol	eIP_CAMERA_PROTOCOL	IP camera: Streaming protocol:	→ eIP_CAMERA_PROTOCOL (ENUM)
sLocation	STRING	IP camera, only for RTSP: Access to the RTSP stream	Format: 'rtsp://USER:PASSWORD@IP-ADRESSE:PORT/PFAD' Consult the documentation of the IP camera for the required information. Example for an Axis camera: 'rtsp://root:root@192.168.82.90:554/axis-media/media.amp?videocodec=jpeg&resolution=1920x1080';
udiPort	UDINT	IP camera, only for RTP: Network RTP port	Default value: 50004
eEncoding	ENUM	IP camera: Stream coding For RTSP: The coding set here must correspond to the specification in the parameter string sLocation.	→ eIP_CAMERA_ENCODING (ENUM)
tLatency	TIME	IP camera: Streaming latency	Default value: T#10MS
tStreamTimeout	TIME	IP camera: Stream timeout Time delay after interruption of the camera stream until the diagnostic message is triggered.	Default value: T#100MS  T0#MS: No timeout configured. The camera image will freeze when the stream stops until the stream is active again.  > T0#MS: Timeout configured. The camera image will freeze when the stream stops until the timeout time has elapsed. The camera window is then displayed in the configured background colour and the message STAT_EVNT_CAMERA_ is displayed. STREAM_INTERRUPTED will be output.

## stCAMERA\_CONTROLS (STRUCT)

Parameter	Data type	Meaning	Possible values	
xStartStop	BOOL	Start / stop camera.	FALSE => TRUE	Start camera. Initial value
			TRUE => FALSE	Stop camera. The set window background is displayed.

## stCAMERA\_IMAGE\_SIZE (STRUCT)

Parameter	Data type	Meaning	Possible values
udiWidth	UDINT	Width of the camera image in pixels.	Initial value = 0 pixels
udiHeight	UDINT	Height of the camera image in pixels.	Initial value = 0 pixels

## stCAMERA\_ROI\_CONFIG (STRUCT)

Parameter	Data type	Meaning	Possible values
stPosition	stWINDOW_POSITION	Position of the Region of Interest (ROI)	→ stWINDOW_POSITION (STRUCT)
stSize	stWINDOW_SIZE	Size of the window of the Region of Interest (ROI)	→ stWINDOW_SIZE (STRUCT)

## stDATE\_TIME (STRUCT)

Designation	Data type	Meaning	Possible values	Default value
uiYear	UINT	Year	2000...2099	2000
uiMonth	UINT	Month	1...12	1
uiDay	UINT	Day	1...31	1
uiHour	UINT	Hour	0...23	0
uiMinute	UINT	Minute	0...59	0
uiSeconds	UINT	Second	0...59	0
eDayOfWeek	→ eDAY_OF_WEEK (ENUM)	Day of the week	→ eDAY_OF_WEEK (ENUM)	0

## stETHERNET\_BRIDGE\_CONFIG (STRUCT)

Designation	Data type	Meaning	Possible values	
xEnable	BOOL	Ethernet bridge mode	FALSE	disable / disabled
			TRUE	enable / enabled

Designation	Data type	Meaning	Possible values
stIPv4Setting	→ stIPv4SETTING (STRUCT)	Ethernet IPv4 settings of the Ethernet bridge	→ stIPv4SETTING (STRUCT)
eInterfaceA	→ eETHERNET_INTERFACES (ENUM)	1. Ethernet interface	→ eETHERNET_INTERFACES (ENUM)
eInterfaceB	→ eETHERNET_INTERFACES (ENUM)	2. Ethernet interface	→ eETHERNET_INTERFACES (ENUM)

### stIPv4SETTING (STRUCT)

Designation	Data type	Meaning	Possible values	
slpAddress	STRING(15)	IP address of the device (IPv4)	e.g. 192.168.1.10	
sSubnetMask	STRING(15)	Subnet mask	e.g. 255.255.255.0	
sDefaultGateway	STRING(15)	IP address of the network gateway	e.g. 192.168.1.1	
xDHCPStat	BOOL	Status of the DHCP client of the device	FALSE	DHCP client deactivated
			TRUE	DHCP client active

### stKEY\_LED\_CONTROL (STRUCT)

Designation	Data type	Meaning	Possible values
stKey0_0	→ stLED_SETTINGS (STRUCT)	LED group 0, key 0	→ stLED_SETTINGS (STRUCT)
stKey0_1		LED group 0, key 1	
stKey0_2		LED group 0, key 2	
stKey0_3		LED group 0, key 3	
stKey0_4		LED group 0, key 4	
stKey1_0		LED group 1, key 0	
stKey1_1		LED group 1, key 1	
stKey1_2		LED group 1, key 2	
stKey1_3		LED group 1, key 3	
stKey1_4		LED group 1, key 4	
stKey2_Navigation		LED navigation keys	

### stLED\_SETTINGS (STRUCT)

Parameter	Data type	Meaning	Possible values	
xON	BOOL	Activate/deactivate night mode key LED illumination	TRUE	Activate night mode brightness
			FALSE	Deactivate night mode brightness, activate normal brightness (default value)

Parameter	Data type	Meaning	Possible values
usiBrightness	USINT	<p>Brightness of the key LEDs in night mode.</p> <p>When night mode is activated, the RGB colour values will be reduced to the set percentage.</p> <p>Example:  key LEDs RGB in day mode = WHITE = (00, 255, 255, 255)  Key LED RGB in night mode (brightness = 20 %) = WHITE = (00,51,51,51)</p>	<p>1...100 %</p> <p>Default value = 20 %</p>

### stKEY\_MAP\_LIST (STRUCT)

Parameter	Data type	Meaning	Default value
eKey0_0	eKEY_MAP_CODES (ENUM)	Group 0, key 0	eKEY_MAP_CODES.F1
eKey1_0		Group 1, key 0	eKEY_MAP_CODES.F2
eKey0_1		Group 0, key 1	eKEY_MAP_CODES.F3
eKey1_1		Group 1, key 1	eKEY_MAP_CODES.F4
eKey0_2		Group 0, key 2	eKEY_MAP_CODES.F5
eKey1_2		Group 1, key 2	eKEY_MAP_CODES.F6
eKey0_3		Group 0, key 3	eKEY_MAP_CODES.F7
eKey1_3		Group 1, key 3	eKEY_MAP_CODES.F8
eKey0_4		Group 0, key 4	eKEY_MAP_CODES.F9
eKey1_4		Group 1, key 4	eKEY_MAP_CODES.F10
eKey2_Up		Group 2, key 0, navigation button up	eKEY_MAP_CODES.UP
eKey2_Down		Group 2, key 1, navigation button down	eKEY_MAP_CODES.DOWN
eKey2_Left		Group 2, key 2, navigation button left	eKEY_MAP_CODES.LEFT
eKey2_Right		Group 2, key 3, navigation button right	eKEY_MAP_CODES.RIGHT
eKey2_Enter		Group 2, key 4, navigation button Enter/Return	eKEY_MAP_CODES.RETURN

### stLED\_SETTINGS (STRUCT)

Parameter	Data type	Meaning	Possible values	
xON	BOOL	Switch on/off LED	TRUE	Switch LED on
			FALSE	Switch off LED (default value)
udiColor	UDINT	RGB colour value for the LED	<p>RGB: f 16#00RRGGBB with  RR = 0...FF  GG = 0...FF  BB = 0...FF  Examples:  red = 16#00FF0000  green = 16#0000FF00  blue = 16#000000FF  black = 16#00000000  white = 16#00FFFFFF</p>	

**stLOCAL\_IO (STRUCT)**

Designation	Data type	Meaning	Possible values	
uiAmbientLight	UINT	Ambient light intensity, measured by the integrated light sensor (→ FB GetLightSensor)	0...100 % Default value 0 %	
xInput_00	BOOL	Digital input condition 0 (→ FB GetLocalInputs)	FALSE	Off (default value)
			TRUE	on
xInput_01	BOOL	Digital input condition 1 (→ FB GetLocalInputs)	FALSE	Off (default value)
			TRUE	on
xDiagInput_00	BOOL	Digital output diagnostic feedback condition 0 (→ FB SetLocalOutputs)	FALSE	Off
			TRUE	on
xDiagInput_01	BOOL	Digital output diagnostic feedback condition 1 (→ FB SetLocalOutputs)	FALSE	Off
			TRUE	on
xOutput_00	BOOL	Digital output condition 0 (→ FB SetLocalOutputs)	FALSE	Off
			TRUE	on
xOutput_01	BOOL	Digital output condition 1 (→ FB SetLocalOutputs)	FALSE	Off
			TRUE	on
stSystemTemperatures	→ stSYSTEM_TEMPERATURES (STRUCT)	System temperatures (→ FB GetTemperature)	→ stSYSTEM_TEMPERATURES (STRUCT)	
stSystemVoltages	→ stSYSTEM_VOLTAGES (STRUCT)	System voltages (→ FB GetVoltages)	→ stSYSTEM_TEMPERATURES (STRUCT)	

**stNTP\_SETTINGS**

Element	Data type	Description	Possible values	
xEnable	BOOL	Status and command to enable / disable the NTP service.	FALSE	Default value disable / disabled
			TRUE	enable / enabled
usiServerCnt	USINT	Valid NTP server in aServerList	0...11; Default value = 0	
aServerList	ARRAY [0..10] OF STRING	List of NTP server addresses (maximum 11).	IP addresses or server names of the NTP servers.	

**stOUTPUT\_COMMANDS (STRUCT)**

Parameter	Data type	Meaning	Possible values	
xValue	BOOL	Set condition of the digital output	TRUE	Activate output
			FALSE	disable output (default value)
xErrorReset	BOOL	Reset the error if xError is enabled at the output function block.	FALSE => TRUE	Reset error
			FALSE	(Default value)

## stPDF\_CONTROLS (STRUCT)

Designation	Data type	Meaning	Possible values	
xOpenClose	BOOL	Open or close the PDF document in the PDF viewer window.	TRUE	Open PDF document in PDF viewer window. (Default value)
			FALSE	Close PDF document in the PDF viewer window.
xGotoFirstPage	BOOL	Display the first page of the PDF.	FALSE => TRUE	Execute action
			FALSE	-
xGotoLastPage	BOOL	Show the last page of the PDF.	FALSE => TRUE	Execute action
			FALSE	-
uiGotoPage	UINT	Number of the page to be displayed. The value becomes active in case of a rising edge at xGotoSetPage .	1 (first page)	
xGotoSetPage	BOOL	Display the PDF page set at uiGotoPage.	FALSE => TRUE	Execute action
			FALSE	-
xOnePageDown	BOOL	Scroll down one page.	FALSE => TRUE	Execute action
			FALSE	-
xOnePageUp	BOOL	Scroll up one page.	FALSE => TRUE	Execute action
			FALSE	-
xRotateCCW	BOOL	Rotate PDF view counterclockwise.	FALSE => TRUE	Execute action
			FALSE	-
xRotateCW	BOOL	Rotate PDF view clockwise.	FALSE => TRUE	Execute action
			FALSE	-
xScrollDown	BOOL	Scroll down.	FALSE => TRUE	Execute action
			FALSE	-
xScrollUp	BOOL	Scroll up.	FALSE => TRUE	Execute action
			FALSE	-
sSearchString	STRING	Search tex.	-	
xSearchHere	BOOL	Search for search text in the PDF and jump to the first occurrence of the view text.	FALSE => TRUE	Execute action
			FALSE	-
xSearchDown	BOOL	Search for the next occurrence of the search text in the PDF.	FALSE => TRUE	Execute action
			FALSE	-
xSearchUp	BOOL	Search for the previous occurrence of the search text in the PDF.	FALSE => TRUE	Execute action
			FALSE	-
xToggle_TOC	BOOL	Show or hide the table of contents.	TRUE	show
			FALSE	hide
uiZoom	UINT	Zoom factor in %. The value becomes active with a rising edge at xZoomToValue .	10...1600 %	
xZoomToValue	BOOL	Set the zoom to the set zoom factor on uiZoom .	FALSE => TRUE	Execute action
			FALSE	-
xZoomIn	BOOL	Zoom in	Standard zoom increments: 12, 25, 33, 50, 66, 75, 100, 125, 150, 200, 400, 800, 1600 %	
xZoomOut	BOOL	Zoom out		
xZoomToFitWidth	BOOL	Zoom to window width.	FALSE => TRUE	Execute action
			FALSE	-

**stSYSTEM\_TEMPERATURES (STRUCT)**

Designation	Data type	Meaning	Possible values
rCore0	REAL	Process temperature in °C	e.g. 40.1 °C
rBoard	REAL	Main board temperature in °C	e.g. 43.9 °C

**stSYSTEM\_VOLTAGES (STRUCT)**

Designation	Data type	Meaning	Possible values
rVBB0	REAL	System voltage VBB0 in V	e.g. 28.2 V
rVBB15	REAL	System voltage terminal 15 in V	
rVBB30	REAL	System voltage terminal 30 in V	

**stSYS\_INFO\_BASIC (STRUCT)**

Element	Data type	Description	Possible values	
sDevice_FWver	STRING	Device firmware version	e.g. 1.0.0.0	
sDeviceHWver	STRING	Hardware version	e.g. 1.1.1.1	
sDevice_SerialNum	STRING	Serial number of the device	e.g. 000000017427	
sDevice_ArticleNum	STRING	ifm article number	e.g. CR1203	
	STRING	ifm article name	e.g. ecomatDisplay/7"/Basic	
sDevice_ArticleRev	STRING	ifm article revision	e.g. AB	
xTouchScreen_Available	BOOL	Touch screen functionality available yes / no	FALSE	No touch screen available
			TRUE	Touch screen available

**stSYS\_INFO\_EXTENDED (STRUCT)**

Element	Data type	Description	Possible values	
uiNumOfETHitf	UINT	Number of available Ethernet interfaces	e.g. 2	
uiNumOfUSBItf	UINT	Number of available USB interfaces	e.g. 1	
usiNumOfACam	USINT	Number of available analogue camera input streams	e.g. 1	
usiLocalIO_NumOfDI	USINT	Number of available digital inputs	e.g. 2	
usiLocalIO_NumOfDO	USINT	Number of available digital outputs	e.g. 2	
usiLocalIO_NumOfStatLED	USINT	Number of available status LEDs on the front panel	e.g. 1	
usiLocalIO_NumOfLightSensors	USINT	Number of available light sensors on the front panel	e.g. 1	
xAudio_HeadPhoneOutAvailable	BOOL	Availability of the headphone output channel	FALSE	not available
			TRUE	available
xAudio_SpeakerOutAvailable	BOOL	Availability of the loudspeaker output channel	FALSE	not available
			TRUE	available

Element	Data type	Description	Possible values	
xAudio_LineInAvailable	BOOL	Availability of the LineIn input channel	FALSE	not available
			TRUE	available
sDevice_MfgDate	STRING	Manufacturing date	In the format DD.MM.YYYY, hh:mm:ss, e.g.. 01.12.2019, 09:13:55	
sDevice_MAC_ETHItf_0	STRING	MAC address of Ethernet interface ETH0	e.g. 00:01:02:06:63:DE	
sDevice_MAC_ETHItf_1	STRING	MAC address of Ethernet interface ETH1	e.g. 00:01:02:06:63:DF	
usiNumOfKeypad	USINT	Number of available keypads	e.g. 1	
usiKeypad_0_NumOfKeys	USINT	Number of keys on keypad 0	e.g. 6	
usiKeypad_0_NumOfNaviElem	USINT	Number of navigation keys on keypad 0	e.g. 5	
usiKeypad_1_NumOfKeys	USINT	Number of keys on keypad 1	e.g. 6	
usiKeypad_1_NumOfNaviElem	USINT	Number of navigation keys on keypad 1	e.g. 5	
uiLCD_Width_mm	UINT	Width of the LCD in mm	e.g. 153	
uiLCD_Height_mm	UINT	Height of the LCD in mm	e.g. 92	
uiLCD_Width_Pixels	UINT	Width of the LCD in pixels	e.g. 800	
uiLCD_Height_Pixels	UINT	Height of the LCD in pixels	e.g. 480	
usiLCD_ColourDepth_bpp	USINT	Colour depth of the LCD in bpp	e.g. 8 = 8 bpp	
usiLCD_DiagonalSize_Inch	USINT	image diagonal of the LCD in inches	e.g.. 7 = 7 inches	

## stSYS\_INFO\_INTERNAL (STRUCT)

Element	Data type	Description	Possible values
sDevice_ProductType	STRING	Product type	e.g. pdm3_12w_002
udiDevice_SWcompatibility	UDINT	Software compatibility	e.g. 4
sDevice_ProductionOrderNum	STRING	Device production number	e.g. 9999999
sKeypad_0_FWver	STRING	Firmware version keypad 0	
sKeypad_0_HWver	STRING	Hardware version keypad 0	
sKeypad_1_FWver	STRING	Firmware version keypad 1	
sKeypad_1_HWver	STRING	Hardware version keypad 1	
udiFrontPanel_SerialNum	UDINT	Front panel serial number	e.g. 4294967296
sFrontPanel_MfgWeek	STRING	Front panel week of manufacture	e.g. 51
sFrontPanel_MfgYear	STRING	Front panel year of manufacture	e.g. 2019
uiFrontPanel_VendorId	UINT	Front panel manufacturer ID	e.g. 65535
uiFrontPanel_Id	UINT	Front panel ID	e.g. 65535
sLCD_MfgrName	STRING	LCD manufacturer name	
usiTouchScreen_Address	USINT	Touch screen address	e.g. 5A
uiTouchScreen_VendorId	UINT	Touch screen controller manufacturer ID	e.g. 65535-
usiTouchScreen_DeviceId	USINT	Touch screen controller device ID	e.g. 65535
sMainBoard_OrderNum	STRING	Production number of the mainboard	e.g. 9999999
sMainBoard_MaterialNum	STRING	ERP material number of the main board	e.g. 9999999



Element	Data type	Description	Possible values
sMainBoard_MaterialRev	STRING	ERP material revision of the main board	e.g. 00

### stUSB\_STORAGE\_INFO (STRUCT)

Element	Data type	Description	Possible values	
sFileSystemName	STRING(40)	Name of the mounted file system	z.B. /dev/sda1 Default value: "not available"	
sAccessPath	STRING(50)	USB memory directory	e.g. /media/usb/sda1 Default value: "not available"	
sFileSystemType	STRING(16)	File system type	e.g. vfat Default value: "not available"	
xRemove	BOOL	Command to unmount the USB memory	FALSE -> TRUE	Activate unmount.
			FALSE	Unmount operation was executed.

### stWINDOW\_CONFIG (STRUCT)

Parameter	Data type	Meaning	Possible values
stPosition	stWINDOW_POSITION	Window position	→ stWINDOW_POSITION (STRUCT)
stSize	stWINDOW_SIZE	Window size in pixels	→ stWINDOW_SIZE (STRUCT)
sFileName_PrintWindow	STRING (255)	File name, path and format for screen shot.	e.g.. '/home/cds-apps/ScreenShot.jpg'

### stWINDOW\_CONTROLS (STRUCT)

Parameter	Data type	Meaning	Possible values
xVisible	BOOL	Show / hide window	FALSE => TRUE The window is displayed: Initial value
			TRUE => FALSE Window disappears.
xBringToTop	BOOL	Move window to the foreground.	FALSE => TRUE Move window to the foreground.
			FALSE
xSendToBottom	BOOL	Move window to the background.	FALSE => TRUE Move window to the background.
			FALSE
xOneLevelUp	BOOL	Move window one level forward.	FALSE => TRUE Move window one level forward.
			FALSE
xOneLevelDown	BOOL	Move window one level backward.	FALSE => TRUE Move window one level backward.
			FALSE
xLockPosition	BOOL	Lock / unlock window position.	FALSE => TRUE Lock window position.

Parameter	Data type	Meaning	Possible values	
xLockPosition	BOOL	Lock / unlock window position.	TRUE => FALSE	Unlock window position.
xLockSize	BOOL	Lock / unlock window size.	FALSE => TRUE	Lock window size.
			TRUE => FALSE	Unlock window size.
xPrintWindow	BOOL	Create a screenshot of the window content and save it as a file.	FALSE => TRUE	Create a screenshot.
			FALSE	

### stWINDOW\_POSITION (STRUCT)

Parameter	Data type	Meaning	Possible values
uiX	UINT	X position of the window in pixels.	0...4096; Initial value = 0
uiY	UINT	Y position of the window in pixels.	0...4096; Initial value = 0

### stWINDOW\_SIZE (STRUCT)

Parameter	Data type	Meaning	Possible values
uiWidth	UINT	Width of the window in pixels.	0...4096; Initial value = 0 pixels
uiHeight	UINT	Height of the window in pixels.	0...4096; Initial value = 0 pixels

## 9.4.16 Global

### aETH\_ITF\_LIST (ALIAS)

List of the names of all available Ethernet interfaces.

### aTIME\_ZONE\_LIST (ALIAS)

Name	Description	Data type	Value
Time zone 0	Designation RTC time zone 0	STRING(31)	0
Time zone 1	Designation RTC time zone 1	STRING(31)	1
...	...	STRING(31)	...
Time zone 120	Designation RTC time zone 120	STRING(31)	120

### aUSB\_STORAGE\_INFO\_LIST (ALIAS)

List of the mounted USB memory devices.

### GCL\_Data (GVL)

Name	Description	Data type	Value
usiMaxEthItf	Number of Ethernet interfaces in the device	USINT	16
usiNumberOfSysInfo	Number of system information parameters of the device.	USINT	7
usiMaxStrgDevi	Number of supported USB storage media.	USINT	16
usiMaxTimeZones	Number of RTC time zones.	USINT	121

## 9.5 ifmFileUtil.library

The library contains program blocks (POU), data structures and enumeration types for file operations.

### 9.5.1 Generic File

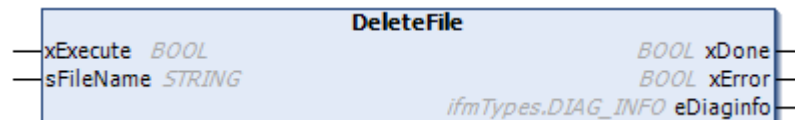
#### DeleteFile

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmFileUtil.library

**Symbol in CODESYS:**



#### Description

The FB deletes a file from the FLASH memory of the device or the USB memory device.

#### Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
sFileName	STRING(255)	Directory path and name of the file	e.g. '/home/project/data.txt'	



The following entries for "sFileName" are invalid and cause an error message:

- Value contains blanks
- No value entered
- Value is a file (e. g. /home/cds-apps/)
- Value contains subsequent "/" (e. g. /home/cds-apps///LogFile.csv)

#### Output parameter

Parameter	Data type	Meaning	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

Parameter	Data type	Meaning	Possible values
xError	BOOL	Indication if an error occurred during the FB execution	TRUE <ul style="list-style-type: none"> <li>An error has 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                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                         Status: 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\_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!

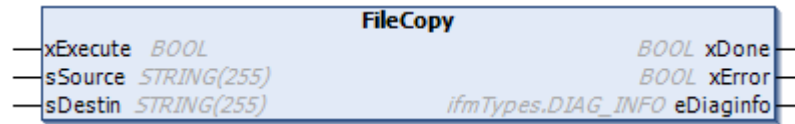
## FileCopy

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmFileUtil.library

**Symbol in CODESYS:**



## Description

The FB copies one or several files from a source directory on the device to a destination directory on the device.



The FB uses the Linux command "cp -rf <sSource> <sDestin>".

## Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
sSource	STRING	Source directory / file	e.g. /data/source /data/sourcefile.txt /data/*.txt	
sDestin	STRING	Target directory/file on the device	e.g. /data/destin	



There are the following options for file selection:

- Individual files (e. g. /home/values/sample.csv)
- Several files by means of wildcards (e. g. /home/values/\*.csv)
- All files of a directory (e. g. /home/values/)

The following combinations of source and target indications are valid:

- Values for source and target differ and both are folders
- Values for source and target differ and both are files
- Value for source is file and value for target is folder

The following combinations of source and target indications are invalid and cause an error message:

- No values for source and/or target
- Value for source is folder and value for target is file
- Values for source and/or target contain subsequent "/" (e. g. /home/values///file.csv)
- Values for source and/or target contain blanks
- Values for source and target are identical (file and folder)

## Output parameter

Parameter	Data type	Meaning	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>• An error has 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                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                         Status: 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\_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!

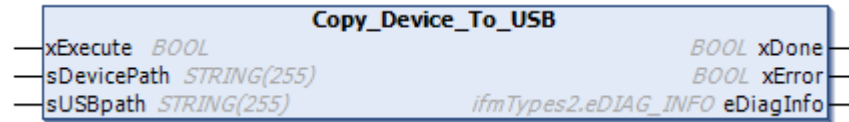
## Copy\_Device\_To\_USB

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmFileUtil.library

**Symbol in CODESYS:**



## Description

The FB copies one or several files from a source directory on the device to a destination directory on an USB memory device.



► The function block uses the Linux command "cp -rf <sDevicePath> <sUSBpath>".

## Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
sDevicePath	STRING	Source directory / file on the device whose content is to be copied	e.g. '/data/tmp/logfile.csv'	
sUSBpath	STRING	Target directory on the USB memory device into which the files are to be copied <ul style="list-style-type: none"><li>Enter the directory as relative path to the mount point of the USB device.</li><li>Example: '/sda1/NewFolder' copies the selected files into the directory '/media/USB/sda1/NewFolder/'</li></ul>	e.g. '/sda1/NewFolder'	





There are the following options for file selection:

- Individual files (e. g. /home/values/sample.csv)
- Several files by means of wildcards (e. g. /home/values/\*.csv)
- All files of a directory (e. g. /home/values/)

The following combinations of source and target indications are valid:

- Values for source and target differ and both are folders
- Values for source and target differ and both are files
- Value for source is file and value for target is folder

The following combinations of source and target indications are invalid and cause an error message:

- No values for source and/or target
- Value for source is folder and value for target is file
- Values for source and/or target contain subsequent "/" (e. g. /home/values///file.csv)
- Values for source and/or target contain blanks
- Values for source and target are identical (file and folder)

## Output parameter

Parameter	Data type	Meaning	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>• An error has 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     | Status: FB/Function is inactive.  |
| • STAT_BUSY         | Status: FB/Function is currently executed.  |
| • STAT_DONE         | Status: 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_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<br>Contact the ifm Service Center!   |
| • ERR_UNDEFINED     | Error: Unknown error<br>Contact the ifm Service Center!   |

## Copy\_USB\_To\_Device

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmFileUtil.library

**Symbol in CODESYS:**



## Description

The FB copies one or several files from a USB memory device to a destination directory on the device.



► The function block uses the Linux command "cp -rf < sUSBpath > < sDevicePath >".

## Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
sUSBpath	STRING	Source directory / file on the USB memory device <ul style="list-style-type: none"><li>Enter the directory as relative path to the mount point of the USB device.</li><li>Example: 'sda1/NewFolder' copies all files of the directory ' /media/USB/sda1/NewFolder/'</li></ul>	e.g. ' /sda1/NewFolder '	
sDevicePath	STRING	Target directory on the device	e.g. ' /data/tmp/ '	



There are the following options for file selection:

- Individual files (e. g. /home/values/sample.csv)
- Several files by means of wildcards (e. g. /home/values/\*.csv)
- All files of a directory (e. g. /home/values/)

The following combinations of source and target indications are valid:

- Values for source and target differ and both are folders
- Values for source and target differ and both are files
- Value for source is file and value for target is folder

The following combinations of source and target indications are invalid and cause an error message:

- No values for source and/or target
- Value for source is folder and value for target is file
- Values for source and/or target contain subsequent "/" (e. g. /home/values///file.csv)
- Values for source and/or target contain blanks
- Values for source and target are identical (file and folder)

## Output parameter

Parameter	Data type	Meaning	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>• An error has 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     | Status: FB/Function is inactive.  |
| • STAT_BUSY         | Status: FB/Function is currently executed.  |
| • STAT_DONE         | Status: 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_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<br>Contact the ifm Service Center!   |
| • ERR_UNDEFINED     | Error: Unknown error<br>Contact the ifm Service Center!   |

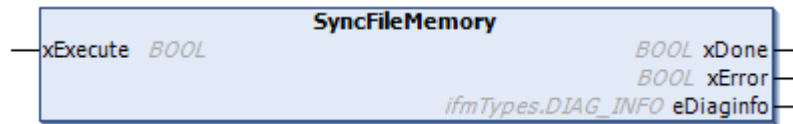
## SyncFileMemory

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmFileUtil.library

**Symbol in CODESYS:**



## Description

The FB synchronises the NAND flash memory with the working memory (RAM) of the device.



► Carry out the FB as last operation before shutting down the device!



► The FB uses the Linux command "sync".

## Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing

## Output parameter

Parameter	Data type	Meaning	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>• An error has 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	Status: FB/Function is inactive.
• STAT_BUSY	Status: FB/Function is currently executed.
• STAT_DONE	Status: 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_UNDEFINED	Error: Unknown error Contact the ifm Service Center!

## 9.5.2 Parameter File

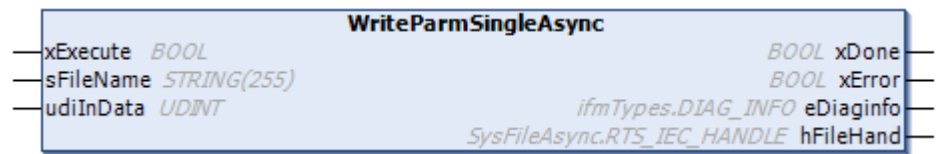
### WriteParmSingleAsync

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:



### Description

The FB writes a single parameter of the UDINT type into a text file.

The parameter is stored in the file as a 10-digit value and right-aligned.

Example:

Value	Saved in the file as
1	0000000001
123	0000000123
1234567890	1234567890

### Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
sFileName	STRING(255)	Directory path and name of the file	e.g. '/home/project/data.txt'	
udiInData	UDINT	Parameter to be written	0 ... 65535	

### Output parameter

Parameter	Data type	Meaning	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

Parameter	Data type	Meaning	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	TRUE	<ul style="list-style-type: none"> <li>An error has occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
hFileHand	RTS_IEC_HANDLE	File description of the runtime system	< 1	Error
			Other	no error

#### Diagnostic codes:

- |                     |   |
|---------------------|---|
| • STAT_INACTIVE     | Status: FB/Function is inactive.  |
| • STAT_BUSY         | Status: FB/Function is currently executed.  |
| • STAT_DONE         | Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.             |
| • ERR_INVALID_VALUE | Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped. |
| • ERR_FILE_SYSTEM   | Error: File operation failed.   |
| • ERR_INTERNAL      | Error: Internal system error<br>Contact the ifm Service Center!   |
| • ERR_NO_OBJECT     | Error: File not available.  |

## ReadParmSingleAsync

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:



## Description

The FB reads a single data record of the UINT type of a text file and provides it.

The file must only contain this single data record. The value in the file must be stored as a 10-digit value and right-aligned.

Example:

Value	Saved in the file as
1	000000001
123	000000123
12345567890	1234567890

## Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
sFileName	STRING(255)	Directory path and name of the file	e.g. '/home/project/data.txt'	

## Output parameter

Parameter	Data type	Meaning	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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
hFileHand	RTS_IEC_HANDLE	File description of the runtime system	< 1	Error



Parameter	Data type	Meaning	Possible values	
hFileHand	RTS_IEC_HANDLE	File description of the runtime system	Other	no error
udiReadParam	UDINT	Parameter read from the text file	0 ... 4294967295	

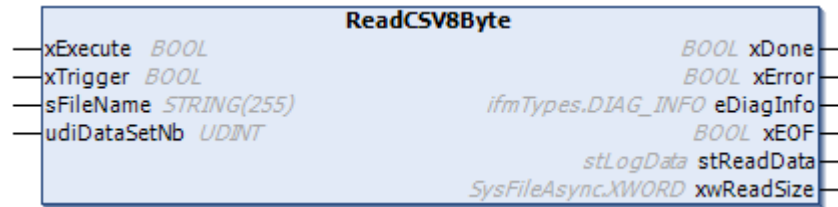
#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_FILE\_SYSTEM                 Error: File operation failed.
- ERR\_INTERNAL                      Error: Internal system error  
Contact the ifm Service Center!
- ERR\_NO\_OBJECT                    Error: File not available.

### 9.5.3 Log File

#### ReadCSV8Byte

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE with Trigger  
**Library:** ifmFileUtil.library  
**Symbol in CODESYS:**



#### Description

The FB reads data records of a CSV file and provides them. The FB assigns the value 0 to unused cells. The FB stores the time stamp of each data record. The CSV must have been created using the WriteCSV8Byte (→ 238).



Each data record has to have a size of 54 bytes.

#### Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
xTrigger	BOOL	Trigger action	FALSE => TRUE	FB reads the selected data record
			Other	no action
sFileName	STRING(255)	Directory path and name of the file	e.g. '/home/project/data.txt'	
udiDataSetNb	UDINT	Number of the data record to be read from the file	0 ... 65535	

#### Output parameter

Parameter	Data type	Meaning	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

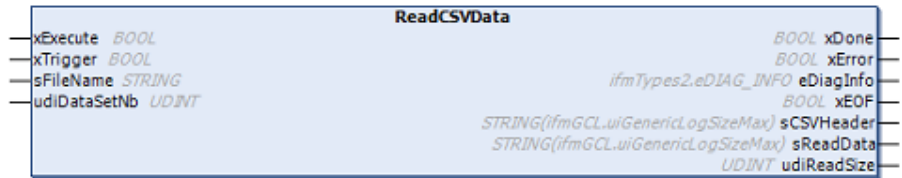
Parameter	Data type	Meaning	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	TRUE	<ul style="list-style-type: none"> <li>An error has occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xEOF	BOOL	indication of the file end	FALSE	file end not yet reached
			TRUE	file end reached
stReadData	stLogData	Data structure with all read data → stLogData (STRUCT)		
xwReadSize	XWORD	Size of the data record read last	54	Reading process successful
			Other	Reading process faulty

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_READY                        Status: File ready for triggered write / read accesses
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_FILE\_SYSTEM                 Error: File operation failed.
- ERR\_NO\_OBJECT                    Error: File not available.

## ReadCSVData

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE with Trigger  
**Library:** ifmFileUtil.library  
**Symbol in CODESYS:**



## Description

The FB reads data records of a CSV file and provides them.

The FB assigns the value 0 to unused cells. The FB stores the time stamp of each data record.

The CSV file must have been created using the FB WriteCSVData\_Linear (→ 243) or the FB WriteCSVData\_Ring (→ 246).

## Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB opens file
			TRUE => FALSE	<ul style="list-style-type: none"> <li>FB closes file</li> <li>All outputs are reset</li> </ul>
xTrigger	BOOL	Trigger action	FALSE => TRUE	FB reads the selected data record
			Other	no action
sFileName	STRING	Directory path and name of the file	e.g. '/home/project/data.csv'	
udiDataSetNb	UDINT	Number of the data record to be read from the file	0 ... 4294967295	

## Output parameter

Parameter	Data type	Meaning	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>An error has occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xEOF	BOOL	indication of the file end	FALSE	file end not yet reached
			TRUE	file end reached

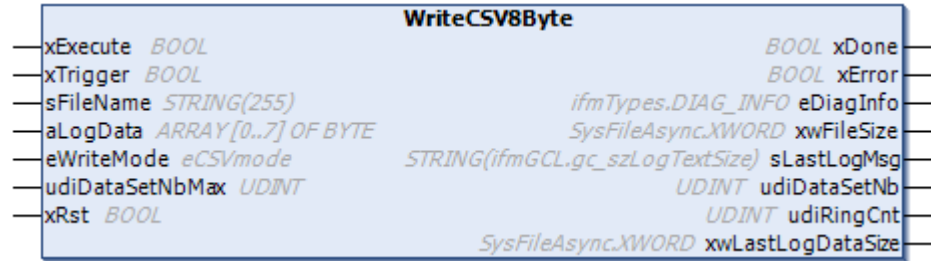
Parameter	Data type	Meaning	Possible values
sCSVHeader	STRING	CSV header read from the file.	CSV header: if there is one in the file
			First record: if there is no CSV header in the file.
sReadData	STRING	CSV record with all data read from the file	--
udiReadSize	UDINT	Size of the last read data record in bytes	--

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_READY                        Status: File ready for triggered write / read accesses
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_RESET                        Status: FB/function executes a RESET operation.
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_FILE\_SYSTEM                 Error: File operation failed.
- ERR\_NO\_OBJECT                    Error: File not available.
- ERR\_UNDEFINED                    Error: Unknown error  
Contact the ifm Service Center!

## WriteCSV8Byte

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE with Trigger  
**Library:** ifmFileUtil.library  
**Symbol in CODESYS:**



## Description

The FB reads the data of an array of 8 bytes and writes them into a CSV file as a data record. The FB stores a time stamp in addition to each data record (date, time). After the data record has been written the FB automatically increments the pointer to the next array in the data structure.

Principle:

No. of the data record	Content of the data record
1	Time stamp*, byte 0, byte 1, ..., byte 7
2	Time stamp, byte 0, byte 1, ..., byte 7
...	...
udiDataSetNbMax	Time stamp, byte 0, byte 1, ..., byte 7
* ...Format of the time stamp DD.MM.YYYY HH:MM:SS	

Example for a data record:

04.08.2016 19:59:55,0,15,245,15,251,15,0,8

The user can choose between the following write modes:

- **Linear**  
The data records are written linearly. The number of data records is theoretically unlimited. it is only limited by the maximum file size specified by the operating system. Existing data records will not be overwritten.
- **Ring:**  
The data records are written into a ring buffer. The number of data records is limited. After the last memory cell of the ring buffer has been written, the FB starts again at the first memory cell of the ring buffer. Existing data records will be overwritten.



In the "linear" mode, the bytes 0 to 53 are reserved for the CSV header. The CSV header can be added subsequently using the FB WriteCSV8ByteHeader (→ 241).

In the ring mode, the CSV header must be written into the file before writing the data records.

## Input parameter

Parameter	Data type	Meaning	Possible values
xExecute	BOOL	Control execution of the FB	FALSE => TRUE FB opens file
			TRUE => FALSE • FB closes file • All outputs are reset

Parameter	Data type	Meaning	Possible values	
xTrigger	BOOL	Trigger action	FALSE => TRUE	FB reads the selected data record
			Other	no action
sFileName	STRING(255)	Directory path and name of the file	e.g. '/home/project/data.txt'	
aLogData	ARRAY [0..7] OF BYTE	Array with the data to be written into the CSV file		
eWriteMode	eCSVmode	Write mode → eCSVmode (ENUM)	LINEAR	linear mode
			RING	ring mode
udiDataSetNbMax	UDINT	Maximum number of data records in the ring mode	0 ... 65535	
xRst	BOOL	Delete CSV file and pointer to the data record	FALSE => TRUE	Delete pointer and CSV file
			Other	no action
* ... preset value				



The following entries for "sFileName" are invalid and cause an error message:

- Value contains blanks
- No value entered
- Value is a file (e. g. /home/cds-apps/)
- Value contains subsequent "/" (e. g. /home/cds-apps///LogFile.csv)

## Output parameter

Parameter	Data type	Meaning	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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xwFileSize	XWORD	Current file size of the file (in bytes)	0 ... 4294967295	
sLastLogMsg	STRING	data record written last as STRING	e.g. 06.12.2016 20:35:45,10,101,255,103,104,105,106,\$n	
udiDataSetNb	UDINT	Number of the data record written into the file last	0 ... 65535	

---

Parameter	Data type	Meaning	Possible values
udiRingCnt	UDINT	Counter for completed cycles of the ring buffer storage	0 ... 65535

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_READY                        Status: File ready for triggered write / read accesses
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_FILE\_SYSTEM                 Error: File operation failed.
- ERR\_INTERNAL                      Error: Internal system error  
Contact the ifm Service Center!
- ERR\_NO\_OBJECT                    Error: File not available.



## WriteCSV8ByteHeader

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE with Trigger  
**Library:** ifmFileUtil.library  
**Symbol in CODESYS:**



GB

### Description

The FB writes the header into a CSV file. The header is always written in the bytes 0 to 53 of the file.

### Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
sFileName	STRING(255)	Directory path and name of the file	e.g. '/home/project/data.txt'	
stHeader	stCSVHeader	Structure with header data → stCSVHeader (STRUCT)	Timestamp,R1C2,R1C3,...,R1C9*	
* ... preset value				



The following entries for "sFileName" are invalid and cause an error message:

- Value contains blanks
- No value entered
- Value is a file (e. g. /home/cds-apps/)
- Value contains subsequent "/" (e. g. /home/cds-apps///LogFile.csv)

### Output parameter

Parameter	Data type	Meaning	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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>

---

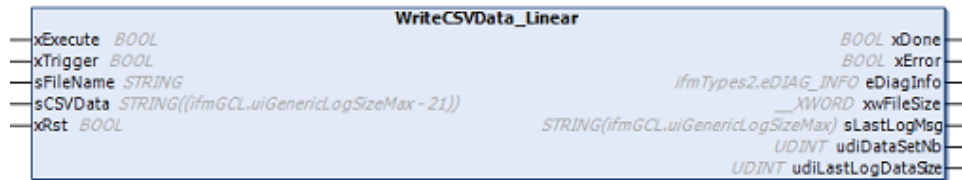
Parameter	Data type	Meaning	Possible values
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)
LastLogMsg	STRING	Written header as STRING	
udiDataSet Posi	UDINT	Number of the data record written last	0...65535

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                          Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_FILE\_SYSTEM                 Error: File operation failed.
- ERR\_INTERNAL                      Error: Internal system error  
Contact the ifm Service Center!
- ERR\_NO\_OBJECT                    Error: File not available.

## WriteCSVData\_Linear

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE with Trigger  
**Library:** ifmFileUtil.library  
**Symbol in CODESYS:**



### Description

The FB writes a data record linearly into a CSV file.

- In case of a rising edge at `xExecute`, the file specified at `sFileName` will be opened.
- If `xExecute` = TRUE, the FB will write the data set `sCSVData` to the open file `sFileName` with each rising edge at `xTrigger`.
- In case of a falling edge at `xExecute`, the file specified at `sFileName` will be closed.
- The FB additionally stores a time stamp in the format `DD.MM.YYYY hh:mm:ss` at the beginning of each data record.
- One data record corresponds to one line in the CSV file.
- The number of data records is theoretically unlimited. it is only limited by the maximum file size set by the operating system or the available memory.
- Existing data records will not be overwritten. New data records are appended to the existing file
- A file header can be written into the CSV file with the FB `WriteCSVHeader` before writing the first data record.



The size of the CSV file increases with each data record written.

The file can become very large.

- ▶ Check the memory capacity of the unit.

Principle:

No. of the data record	Content of the data record
1	DD.MM.YYYY hh:mm:ss, byte 0, byte 1, real 0, string 0, real 1, byte 2
2	DD.MM.YYYY hh:mm:ss, byte 0, byte 1, real 0, string 0, real 1, byte 2
...	...

Examples for a data record:

24.09.2020 19:59:55,0,15,245.45,'Errorcode 996346',251.43567,15

### Input parameter

Parameter	Data type	Meaning	Possible values
xExecute	BOOL	Control execution of the FB	FALSE => TRUE FB opens file

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	TRUE => FALSE	<ul style="list-style-type: none"><li>• FB closes file</li><li>• All outputs are reset</li></ul>
xTrigger	BOOL	Trigger action	FALSE => TRUE	FB reads the selected data record
			Other	no action
sFileName	STRING(255)	Directory path and name of the file	e.g. '/home/project/data.txt'	
sCSVData	STRING ifmGCL (GVL) - 21	CSV data to be saved in the file. Reserved memory for timestamp and new line character at the end of the line (21 bytes)		
xRst	BOOL	Delete CSV file and pointer to the data record	FALSE => TRUE	Delete pointer and CSV file
			Other	no action
* ... preset value				



The following entries for "sFileName" are invalid and cause an error message:

- Value contains blanks
- No value entered
- Value is a file (e. g. /home/cds-apps/)
- Value contains subsequent "/" (e. g. /home/cds-apps///LogFile.csv)

## Output parameter

Parameter	Data type	Meaning	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>An error has occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xwFileSize	XWORD	Current file size of the file (in bytes)	0 ... 4294967295	
sLastLogMsg	STRING	data record written last as STRING	e.g. 06.12.2016 20:35:45,10,101,255,103,104,105,106,\$n	
udiDataSetNb	UDINT	Number of the data record written into the file last	0 ... 65535	

Parameter	Data type	Meaning	Possible values
udiLastLogDataSize	UDINT	Size of the last written data record in bytes.	0 ... ifmGCL (GVL)

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_READY                        Status: File ready for triggered write / read accesses
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_RESET                        Status: FB/function executes a RESET operation.
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_FILE\_SYSTEM                 Error: File operation failed.
- ERR\_LINUX\_SYS\_CALL             Error while executing the Linux command.
- ERR\_ASYNC\_TASK                  Error: asynchronous task provides an error.
- ERR\_NO\_OBJECT                    Error: File not available.
- ERR\_UNDEFINED                    Error: Unknown error  
Contact the ifm Service Center!
- ERR\_GET\_DATE\_TIME                Error: Error while reading date and/or time



Additional information to ERR\_INVALID\_VALUE:

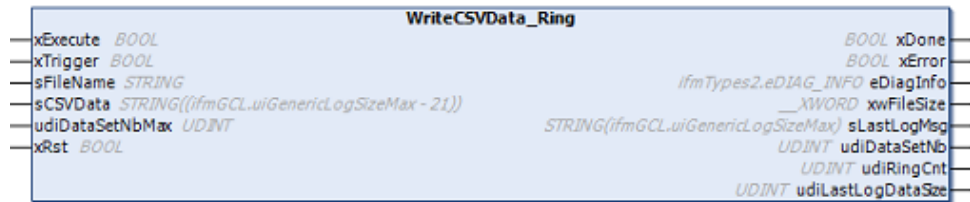
Possible causes at input parameter `sFileName`:

- Invalid directory
- No file path specified
- Invalid sequence of several "/"
- Invalid characters

## WriteCSVData\_Ring

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE with Trigger  
**Library:** ifmFileUtil.library

**Symbol in CODESYS:**



## Description

The FB writes a data record in ring mode into a CSV file.

- In case of a rising edge at `xExecute`, the file specified at `sFileName` will be opened.
- If `xExecute` = TRUE, the FB will write the data set `sCSVData` to the open file `sFileName` with each rising edge at `xTrigger`.
- In case of a falling edge at `xExecute`, the file specified at `sFileName` will be closed.
- The FB additionally stores a time stamp in the format DD.MM.YYYY hh:mm:ss at the beginning of each data record.
- One data record corresponds to one line in the CSV file.
- The number of data records is limited by the value at `udiDataSetNbMax`.
- The oldest existing data record will be deleted after reaching the maximum number. The new data record will be appended to the existing file.
- A file header can be written into the CSV file with the FB WriteCSVHeader before writing the first data record.
- The file size remains the same because older entries will be overwritten. The file size is limited by the number of data records.

Principle:

No. of the data record	Content of the data record
1	DD.MM.YYYY hh:mm:ss, byte 0, byte 1, real 0, string 0, real 1, byte 2
2	DD.MM.YYYY hh:mm:ss, byte 0, byte 1, real 0, string 0, real 1, byte 2
...	...

Examples for a data record:

24.09.2020 19:59:55,0,15,245.45,'Errorcode 996346',251.43567,15

## Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB opens file
			TRUE => FALSE	<ul style="list-style-type: none"> <li>• FB closes file</li> <li>• All outputs are reset</li> </ul>

Parameter	Data type	Meaning	Possible values	
xTrigger	BOOL	Trigger action	FALSE => TRUE	FB reads the selected data record
			Other	no action
sFileName	STRING(255)	Directory path and name of the file	e.g. '/home/project/data.txt'	
sCSVData	STRING ifmGCL (GVL) - 21	CSV data to be saved in the file. Reserved memory for timestamp and new line character at the end of the line (21 bytes)		
udiDataSetNbMax	UDINT	Maximum number of data records in the ring mode	0 ... 65535	
xRst	BOOL	Delete CSV file and pointer to the data record	FALSE => TRUE	Delete pointer and CSV file
			Other	no action

\* ... preset value



The following entries for "sFileName" are invalid and cause an error message:

- Value contains blanks
- No value entered
- Value is a file (e. g. /home/cds-apps/)
- Value contains subsequent "/" (e. g. /home/cds-apps///LogFile.csv)

## Output parameter

Parameter	Data type	Meaning	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>• An error has occurred</li> <li>• Action could not be executed</li> <li>• Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	
xwFileSize	XWORD	Current file size of the file (in bytes)	0 ... 4294967295	
sLastLogMsg	STRING	data record written last as STRING	e.g. 06.12.2016 20:35:45,10,101,255,103,104,105,106,\$n	
udiDataSetNb	UDINT	Number of the data record written into the file last	0 ... 65535	
udiRingCnt	UDINT	Counter for completed cycles of the ring buffer storage	0 ... 65535	

Parameter	Data type	Meaning	Possible values
udiLastLogDataSize	UDINT	Size of the last written data record in bytes.	0 ... ifmGCL (GVL)

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_READY                        Status: File ready for triggered write / read accesses
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_RESET                        Status: FB/function executes a RESET operation.
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_FILE\_SYSTEM                 Error: File operation failed.
- ERR\_LINUX\_SYS\_CALL             Error while executing the Linux command.
- ERR\_ASYNC\_TASK                 Error: asynchronous task provides an error.
- ERR\_NO\_OBJECT                    Error: File not available.
- ERR\_UNDEFINED                    Error: Unknown error  
Contact the ifm Service Center!
- ERR\_GET\_DATE\_TIME               Error: Error while reading date and/or time



Additional information to ERR\_INVALID\_VALUE:

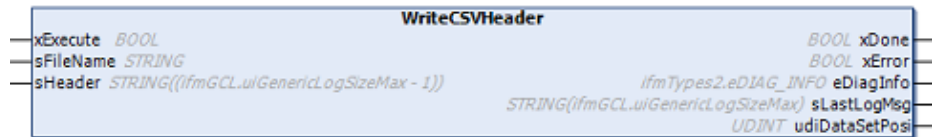
Possible causes at input parameter `sFileName` :

- Invalid directory
- No file path specified
- Invalid sequence of several "/"
- Invalid characters



## WriteCSVHeader

**Function block type:** Function block (FB)  
**Behaviour model:** EXECUTE with Trigger  
**Library:** ifmFileUtil.library  
**Symbol in CODESYS:**



## Description

The FB writes the header line to a CSV file that will then be written with the FB WriteCSVData\_Linear or the FB WriteCSVData\_Ring.

If the file specified at `sFileName` already exists, the beginning of the file is overwritten with the length of the string specified at `sHeader`.

The file will be recreated if it does not already exist.

## Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
sFileName	STRING	Directory path and name of the file	e.g. '/home/project/data.csv'	
sHeader	STRING	String with header data for the CSV file		



The following entries for "sFileName" are invalid and cause an error message:

- Value contains blanks
- No value entered
- Value is a file (e. g. /home/cds-apps/)
- Value contains subsequent "/" (e. g. /home/cds-apps///LogFile.csv)

## Output parameter

Parameter	Data type	Meaning	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

Parameter	Data type	Meaning	Possible values
xError	BOOL	Indication if an error occurred during the FB execution	TRUE <ul style="list-style-type: none"> <li>An error has occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)
sLastLogMsg	STRING	Written header as STRING	
udiDataSetPos	UDINT	Number of the data record written last	0...4294967295

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                          Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_FILE\_SYSTEM                  Error: File operation failed.
- ERR\_NO\_OBJECT                    Error: File not available.
- ERR\_LINUX\_SYS\_CALL              Error while executing the Linux command.
- ERR\_ASYNC\_TASK                   Error: asynchronous task provides an error.
- ERR\_UNDEFINED                    Error: Unknown error  
Contact the ifm Service Center!

## 9.5.4 Support

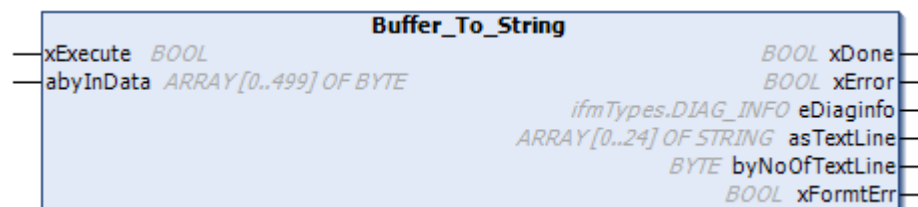
### Buffer\_To\_String

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:



### Description

The FB reads a BYTE array, concatenates the values and provides the result in a STRING array.

- Size of the BYTE array: 500 bytes
- Size of the STRING array: 25 strings

If the FB finds the control character for line end/line break in the source data it writes the subsequent data into a new array line.

If the FB finds a 0 value or the control character for file end in the source data it terminates the conversion.



► BUFFER\_TO\_STRING is a help function for other function blocks.

### Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
abyInData	ARRAY [0...499] OF BYTE	Buffer storage containing the data to be read	per byte: 0x00 ... 0xFF	

### Output parameter

Parameter	Data type	Meaning	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

Parameter	Data type	Meaning	Possible values
xError	BOOL	Indication if an error occurred during the FB execution	TRUE <ul style="list-style-type: none"> <li>An error has occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)
aTextLine	ARRAY [0..24] OF STRING	Array with converted text lines	
byNoOfText Line	BYTE	Number of lines found in the array	

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                         Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.

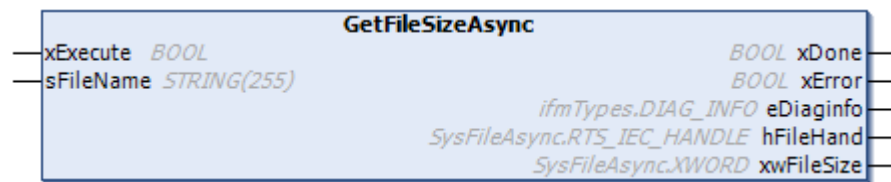
## GetFileSizeAsync

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmFileUtil.library

Symbol in CODESYS:



GB

## Description

The FB determines the size of a file.

## Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
sFileName	STRING(255)	Directory path and name of the file	e.g. '/home/project/data.txt'	



The following entries for "sFileName" are invalid and cause an error message:

- Value contains blanks
- No value entered
- Value is a file (e. g. /home/cds-apps/)
- Value contains subsequent "/" (e. g. /home/cds-apps///LogFile.csv)

## Output parameter

Parameter	Data type	Meaning	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>• An error has occurred</li><li>• Action could not be executed</li><li>• Note diagnostic information</li></ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)	

Parameter	Data type	Meaning	Possible values	
hFileHand	RTS_IEC_HANDLE	File description of the runtime system	< 1	Error
			Other	no error
xwFileSize	XWORD	Current file size of the file (in bytes)	0 ... 4294967295	

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                          Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_FILE\_SYSTEM                  Error: File operation failed.
- ERR\_NO\_OBJECT                    Error: File not available.

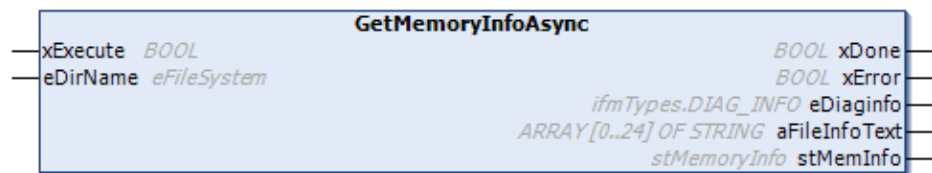
## GetMemoryInfoAsync

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmFileUtil.library

**Symbol in CODESYS:**



## Description

The FB provides detailed information about the memory usage and the available memory of a directory (e. g. /data). The FB stores the information in a text file that has the same name as the value at "eDirName". The text files are stored in the following directories:

eDirName	Storage folder
NAND_FLASH_1GB	/data/
Other	/home/cds-apps/PlcLogic/

Example:

eDirName: HOME

Storage location: /home/cds-apps/PlcLogic/home.txt

## Input parameter

Parameter	Data type	Meaning	Possible values	
xExecute	BOOL	Control execution of the FB	FALSE => TRUE	FB is executed once
			Other	No impact on FB processing
eDirName	eFileSystem	Directory whose storage information is to be read	→ eFileSystem (ENUM)	

## Output parameter

Parameter	Data type	Meaning	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

Parameter	Data type	Meaning	Possible values
xError	BOOL	Indication if an error occurred during the FB execution	TRUE <ul style="list-style-type: none"> <li>An error has occurred</li> <li>Action could not be executed</li> <li>Note diagnostic information</li> </ul>
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)
aFileInfoText	ARRAY [0..24] OF STRING	Array with storage information of the directory path	
stMemInfo	stMemoryInfo	read memory information	→ stMemoryInfo (STRUCT)

#### Diagnostic codes:

- STAT\_INACTIVE                      Status: FB/Function is inactive.
- STAT\_BUSY                          Status: FB/Function is currently executed.
- STAT\_DONE                          Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE                Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.
- ERR\_FILE\_SYSTEM                  Error: File operation failed.
- ERR\_NO\_OBJECT                    Error: File not available.
- ERR\_INTERNAL                      Error: Internal system error  
Contact the ifm Service Center!



## 9.5.5 Function

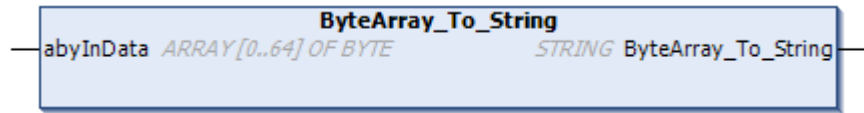
### ByteArray\_To\_String

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



### Description

The function reads a byte array, concatenates the values and provides them as a STRING.

### Input parameter

Parameter	Data type	Meaning	Possible values
abyInData	ARRAY [0...63] OF BYTE	Array with input data	per byte: 0x00 ... 0xFF

### Output parameter

Parameter	Data type	Meaning	Possible values
ByteArray_To_String	STRING	Returned string	

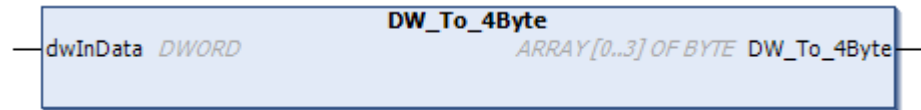
## DW\_To\_4Byte

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



### Description

The function converts a DWORD into an array of 4 bytes.

Example:

DWORD: 0xFFEEDDCC

ARRAY[0..3] OF BYTE: 0xFF | 0xEE | 0xDD | 0xCC

### Input parameter

Parameter	Data type	Meaning	Possible values
dwInData	DWORD	Data	e.g. 0xFFDDEECC

### Output parameter

Parameter	Data type	Meaning	Possible values
DW_To_4BytebyNoOfTextLine	ARRAY [0..3] OF BYTE	Return value of the function	e.g. byte 0: (CC) byte 1: DD byte 2: EE byte 3: FF

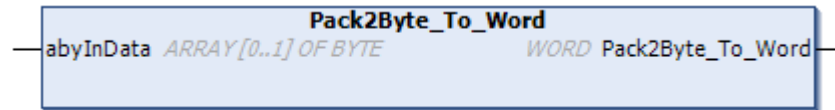
## Pack2Byte\_To\_Word

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



GB

### Description

The function converts an array of 2 bytes into a word.

Example:

alnData[0] = 0x00

alnData[1] = 0xAA

Pack2Byte\_To\_Word = 0xAA00

### Input parameter

Parameter	Data type	Meaning	Possible values
abyInData	ARRAY [0...1] OF BYTE	Array with input data	per byte: 0x00 ... 0xFF

### Output parameter

Parameter	Data type	Meaning	Possible values
Pack2Byte_To_Word	WORD	Return value of the function	0x0000 ... 0xFFFF

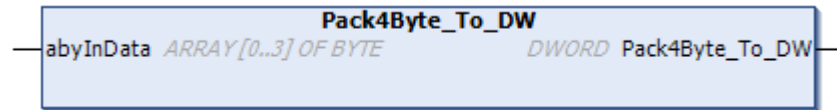
## Pack4Byte\_To\_DW

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



## Description

The function converts an array of 4 bytes into a double word.

Example:

alnData[0] = 0x00

alnData[1] = 0x11

alnData[2] = 0xAA

alnData[3] = 0xFF

Pack4Byte = 0xFFAA1100

## Input parameter

Parameter	Data type	Meaning	Possible values
abyInData	ARRAY [0...3] OF BYTE	Array with input data	per byte: 0x00 ... 0xFF

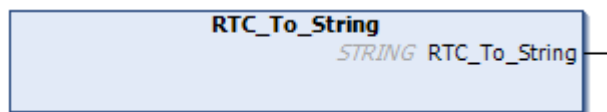
## Output parameter

Parameter	Data type	Meaning	Possible values
Pack4Byte_To_DW	DWORD	Return value of the function	0x00000000 ... 0xFFFFFFFF

---

## RTC\_To\_String

**Function block type:** Function (FUN)  
**Behaviour model:** --  
**Library:** ifmFileUtil.library  
**Symbol in CODESYS:**



GB

### Description

The function reads the device-internal real-time clock (RTC) and provides the operating time in seconds since the last system start as a string.

### Output parameter

Parameter	Data type	Meaning	Possible values
RTC_To_String	STRING	Current time in the format DD.MM.YY hh:mm:ss	E.g. 02.08.2016 08:59:03

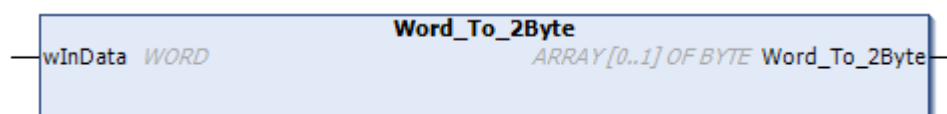
## Word\_To\_2Byte

**Function block type:** Function (FUN)

**Behaviour model:** --

**Library:** ifmFileUtil.library

**Symbol in CODESYS:**



## Description

The function converts a WORD into an array of 2 bytes.

Example:

wInData = 0xFFEE

Word\_To\_2Byte[0] = 0xEE

Word\_To\_2Byte[1] = 0xFF

## Input parameter

Parameter	Data type	Meaning	Possible values
	WORD	Data	0x0000 ... 0xFFFF

## Output parameter

Parameter	Data type	Meaning	Possible values
Word_To_2Byte	ARRAY [0..1] OF BYTE	Return value of the function	Pro byte: 0x00 ... 0xFF

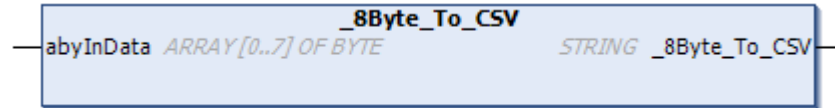
## **\_8Byte\_To\_CSV**

**Function block type:** Function (FUN)

**Behaviour model:** --

**Library:** ifmFileUtil.library

**Symbol in CODESYS:**



**GB**

### **Description**

The function converts the data of an array of 8 bytes into a CSV string and provides this string. One-digit and two-digit numbers are supplemented with leading zeros. The individual values are separated by a comma. The line end is indicated by a comma ( , ).

Example:

```
alnData[0] = 0x00
alnData[1] = 0x01
alnData[2] = 0x63
alnData[3] = 0x64
alnData[4] = 0xA0
alnData[5] = 0xEE
alnData[6] = 0xFF
alnData[7] = 0x11
```

8byte\_To\_CSV = 000,001,099,100,160,238,255,017

### **Input parameter**

Parameter	Data type	Meaning	Possible values
abyInData	ARRAY [0..7] OF BYTE	Array with input data	per byte: 0x00 ... 0xFF

### **Output parameter**

Parameter	Data type	Meaning	Possible values
_8Byte_To_CSV	STRING	CSV string; individual bytes are separated by a comma	e.g. 002004008016032064128255

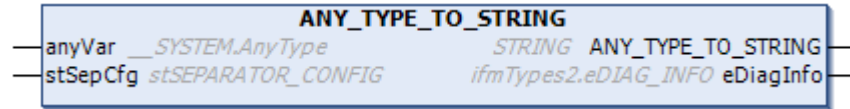
## ANY\_TYPE\_TO\_STRING

Function block type: Function (FUN)

Behaviour model: --

Library: ifmFileUtil.library

Symbol in CODESYS:



### Description

The function converts a data value into a string.

The following data types will be converted: BYTE, WORD, DWORD, LWORD, USINT, UINT, UDINT, ULINT, SINT, INT, DINT, LINT, REAL, LREAL, STRING.

It is possible to define whether a separator is to be added and whether it appears before or after the data value.

### Input parameter

Parameter	Data type	Meaning	Possible values
anyVar	ANY	CSV data to be converted to a string.	-
stSepCfg	stSEPARATOR_CONFIG	CSV separator configuration	stSEPARATOR_CONFIG (STRUCT) (→ 269)

### Output parameter

Parameter	Data type	Meaning	Possible values
ANY_TYPE_TO_STRING	STRING	String with the converted CSV data incl. separator.	--
eDiagInfo	DIAG_INFO	Diagnostic information	→ List below (diagnostic codes)

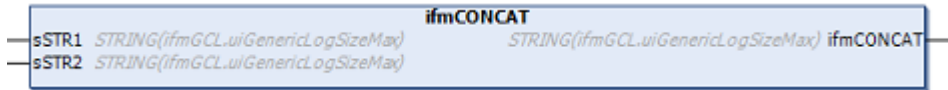
Diagnostic codes:

- STAT\_DONE Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- ERR\_INVALID\_VALUE Error: at least 1 invalid input parameter or invalid combination of input parameters; Function call has been stopped.




ifmCONCAT

Function block type: Function (FUN)  
Behaviour model: --  
Library: ifmFileUtil.library  
Symbol in CODESYS:



Description

The function connects 2 strings and outputs the result in a string.

 ► The string length of the used strings is more than the default 255 characters and is defined in the global constant `ifmGCL.uiGenericLogSizeMax`. ifmGCL (GVL) (→ 269)

Input parameter

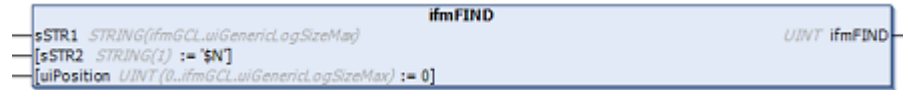
Parameter	Data type	Meaning	Possible values
sStr1	STRING	String 1	--
sStr2	STRING	String 2	--

Output parameter

Parameter	Data type	Meaning	Possible values
ifmCONCAT	STRING	String sStr1 + sStr2	--

## ifmFIND

**Function block type:** Function (FUN)  
**Behaviour model:** --  
**Library:** ifmFileUtil.library  
**Symbol in CODESYS:**



## Description

The function returns the position of the string `sStr2` in the string `sStr1`. The search starts at `uiPosition`.



- The string length of the used strings is more than the default 255 characters and is defined in the global constant `ifmGCL.uiGenericLogSizeMax`. `ifmGCL` (GVL) (→ 269)

## Input parameter

Parameter	Data type	Meaning	Possible values
sStr1	STRING	String 1	--
sStr2	STRING	String 2, this is searched for in string 1.	--
uiPosition	UINT	Start position for searching in the string 1	0 = 1. characters

## Output parameter

Parameter	Data type	Meaning	Possible values
ifmFIND	UINT	Position of sStr2 in sStr1	--

## ifmMID

**Function block type:** Function (FUN)  
**Behaviour model:** --  
**Library:** ifmFileUtil.library  
**Symbol in CODESYS:**

```

ifmMID
---sSTR  STRING(ifmGCL.uiGenericLogSizeMax)  STRING(ifmGCL.uiGenericLogSizeMax) ifmMID
---uiPosition  UINT(0..ifmGCL.uiGenericLogSizeMax)
---uiLength  UINT(0..ifmGCL.uiGenericLogSizeMax)

```

## Description

The function reads a substring of another string.



- The string length of the used strings is more than the default 255 characters and is defined in the global constant `ifmGCL.uiGenericLogSizeMax`. ifmGCL (GVL) (→ 269)

Example:

uiPosition	0	1	2	3	4	5
sStr	a	b	c	d	e	f

sStr = "abcdef"

uiPosition = 1

uiLength = 3

=> ifmMTD = "bcd"

## Input parameter

Parameter	Data type	Meaning	Possible values
sStr	STRING	Character string	--
uiPosition	UINT	Start position in the string	0 = 1. characters
uiLength	UINT	Number of characters	--

## Output parameter

Parameter	Data type	Meaning	Possible values
ifmMID	STRING	Substring with length uiLength from start position uiPosition.	--

## 9.5.6 ENUM

### eCSVmode (ENUM)

Name	Description	Possible values	Data type	Value
eCSVmode	Write mode for CSV file	NONE	INT	0
		LINEAR	INT	1
		RING	INT	2

### eFileSystem (ENUM)

Name	Description	Possible values		Data type	Value
eFileSystem	Directory path for the integration of devices and files	ROOT	/	INT	0
		DEV	/dev	INT	1
		TMPFS	/tmpfs	INT	2
		HOME	/home	INT	3
		NAND_FLASH_1GB	/data	INT	4
		FONTS	/opt/qt-x.y.z/lib/fonts	INT	5
		USB1	/tmpfs/media/usb/<USB Name>	INT	6
		USB2	/tmpfs/media/usb/<USB Name>	INT	7
		USB3	/tmpfs/media/usb/<USB Name>	INT	8
		USB4	/tmpfs/media/usb/<USB Name>	INT	9

## 9.5.7 STRUCT

### stCSVHeader (STRUCT)

Designation	Data type	Meaning	Possible values
sColumn1	STRING(18)	Heading of column 1	DateAndTime*
sColumn2	STRING(3)	Heading of column 2	R1C2*
sColumn3	STRING(3)	Heading of column 3	R1C3*
sColumn4	STRING(3)	Heading of column 4	R1C4*
sColumn5	STRING(3)	Heading of column 5	R1C5*
sColumn6	STRING(3)	Heading of column 6	R1C6*
sColumn7	STRING(3)	Heading of column 7	R1C7*
sColumn8	STRING(3)	Heading of column 8	R1C8*
sColumn9	STRING(3)	Heading of column 9	R1C9*

\* ... \* preset value (R = Row, C = Column, R1C2 = Row 1, Column 2)

### stMemoryInfo (STRUCT)

Designation	Data type	Meaning	Possible values
udiTotalKB	UDINT	Total memory (in Kbytes)	
udiUsedKB	UDINT	Memory used (in Kbytes)	
udiAvailKB	UDINT	Free memory (in Kbytes)	
usiUsedPerc	UDINT	Percentage of the memory used referred to the entire memory (in %)	

**stLogData (STRUCT)**

Designation	Data type	Meaning	Possible values
sTimeStamp	STRING	Timestamp (DD.MM.YYYY HH:MM:SS)	
aData	ARRAY [0...7] OF BYTE	CSV data record	
sRawData	STRING	sTimeStamp and aData as STRING; Values are separated by a comma	

**stLOG\_FILE\_CONFIG (STRUCT)**

Log file parameter.

Designation	Data type	Meaning	Possible values	
xwPosition	__XWORD	Cursor position (byte)		
udiDataSetNb	UDINT	Data record number		
sCSVmode	STRING	CSV file log mode	LINEAR / RING	
udiRingCnt	UDINT	Ring counter; Indicates how often the file has been overwritten in ring mode.		
xwLastLogDataSize	__XWORD	Last logged record size in bytes.		
xHeaderSet	BOOL	Indicates whether a header has been written to the file (FB WriteCSVHeader).	TRUE	Header was written.
			FALSE	Header has not been written yet.
udiDataSetNbMax	UDINT	Maximum number of data records written to the file in RING mode (FIFO).		

**stSEPARATOR\_CONFIG (STRUCT)**

Configuration of the CSV separator.

Designation	Data type	Meaning	Possible values	
sChar	STRING [1]	CSV separator	Comma "," Semicolon ";" (default value) No separator ""	
xAfterData	BOOL	Defines whether the separator is to appear before or after the data value.	TRUE	After the data value, e.g. "1234;"
			FALSE	Before the data value, e.g. ";1234" (default value)

**9.5.8 GlobalConstants****ifmGCL (GVL)**

Name	Description	Data type	Value
uiHeaderSize	Length of the CSV file header (1 byte is reserved for the line break).	UINT	53
uiLogTextSize	Length of the CSV file text (1 byte is reserved for the line break).	UINT	53
sAsyncTaskName	Async task name for the ifmFileUtil library	STRING	'Task_ifmFileUtil'

---

Name	Description	Data type	Value
udiTaskSleepTime	Async task sleep time in ms	UDINT	20
udiTaskTimeout	Async task time out in ms	UDINT	1000
uiGenericLogSizeMax	Length of the CSV data (1 line) including timestamp and new line characters in bytes. A total of 21 bytes are reserved: 20 bytes for the timestamp at the beginning of the line; 1 byte for the line break at the end of the line;	UINT	1024

## 9.6 ifmRawCAN.library

The library contains POU's and data structures for the programming of the CAN Layer 2 level of the CAN interfaces of the device under CODESYS.

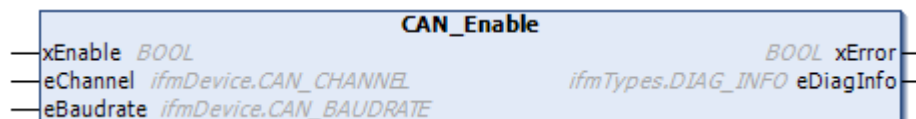
### 9.6.1 CAN\_Enable

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



#### Description

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 parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ CAN_CHANNEL (ENUM)	
eBaudrate	CAN_BAUDRATE	Baud rate of the CAN channel	→ CAN_BAUDRATE (ENUM)	

#### Output parameter

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>An error has 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

Status: FB/Function is inactive.

---

• STAT_DONE	Status: 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!



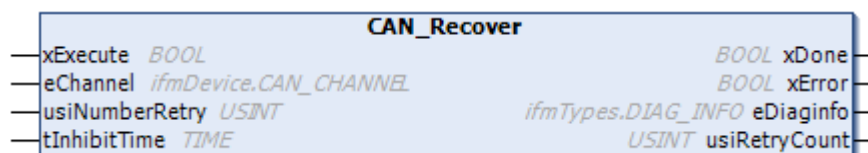
## 9.6.2 CAN\_Recover

**Function block type:** Function block (FB)

**Behaviour model:** EXECUTE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



### Description

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 parameter

Parameter	Data type	Meaning	Possible values
xExecute	BOOL	Control execution of the FB	<ul style="list-style-type: none"> <li>• FALSE =&gt; TRUE: FB is executed once</li> <li>• Otherwise: No impact on FB processing</li> </ul>
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	CAN_CHANNEL (ENUM) (→ □ 204)
	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 parameter

Parameter	Data type	Description	Possible values
xDone	BOOL	Indication of whether execution of the FB has been successfully completed	<ul style="list-style-type: none"> <li>• FALSE: FB is executed</li> <li>• TRUE: FB successfully executed</li> </ul>
xError	BOOL	Indication if an error occurred during the FB execution	<ul style="list-style-type: none"> <li>• FALSE: No error occurred or the FB is still being executed</li> <li>• TRUE: An error has occurred</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	Status: FB/Function is inactive.
• STAT_DONE	Status: 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!

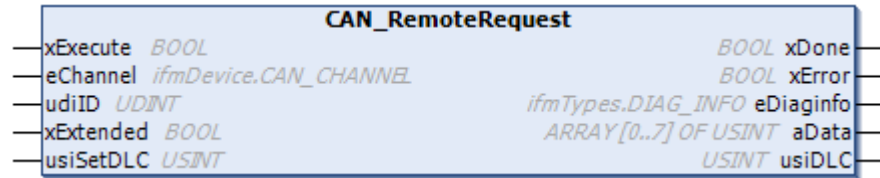
### 9.6.3 CAN\_RemoteRequest

Function block type: Function block (FB)

Behaviour model: EXECUTE

Library: ifmRawCAN.library

Symbol in CODESYS:



#### Description

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 parameter

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	→ CAN_CHANNEL (ENUM)	
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 parameter

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

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	TRUE	<ul style="list-style-type: none"> <li>An error has 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      Status: FB/Function is inactive.
- STAT\_DONE      Status: FB/Function has been successfully executed and completed. There are valid results on the outputs.
- STAT\_BUSY      Status: 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.

## 9.6.4 CAN\_RemoteResponse

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



### Description

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 parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ CAN_CHANNEL (ENUM)	
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
			...	... bytes
			8	8 bytes
* ... preset value				

### Output parameter

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	• An error has occurred • Action could not be executed • Note diagnostic information
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	Status: FB/Function is inactive.
• STAT_DONE	Status: 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!

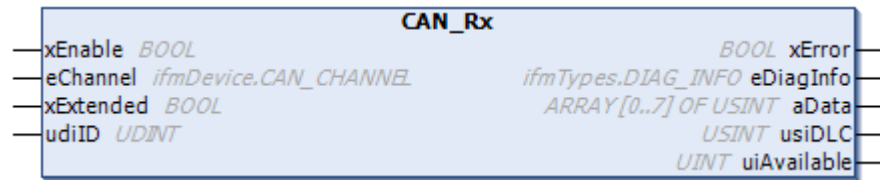
## 9.6.5 CAN\_Rx

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



### Description

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.

### Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ CAN_CHANNEL (ENUM)	
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 parameter

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>An error has 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		

Parameter	Data type	Description	Possible values	
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                      Status: FB/Function is inactive.
- STAT\_DONE                          Status: 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!



## 9.6.6 CAN\_RxMask

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



### Description

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

Form: 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 (→ 279)

### Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ CAN_CHANNEL (ENUM)	
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	eg. 000 0010 0000	
udiIDMask	UDINT	Bit pattern of the required area 1 ... bit relevant for selection 0 ... bit not relevant for selection	eg. 000 1111 1111	

\* ... preset value

## Output parameter

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>An error has 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      Status: FB/Function is inactive.
- STAT\_DONE      Status: 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!

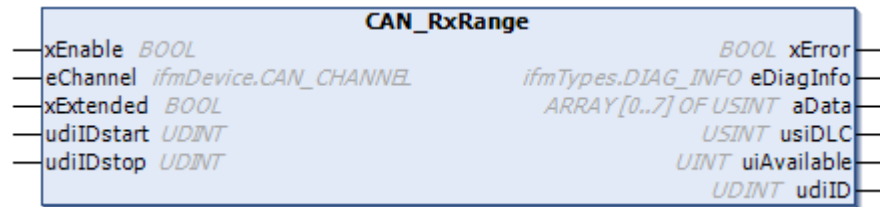
## 9.6.7 CAN\_RxRange

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



### Description

The FB receives CAN messages of a coherent area. The area is defined by an upper and lower limit.

The following rules apply to the definition of this area:

Lower and upper limit:

Standard frames: 0 ... 2047 (11 bit identifier)

Extended frames: 0 ... 536 870 911 (29 bit identifier)

The value for the lower limit must be  $\leq$  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.



General behaviour of the FB: CAN\_Rx (→ 279)

### Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ CAN_CHANNEL (ENUM)	
xExtended	BOOL	Requested frame type: - Standard Frame (11 bits identifier) - Extended-Frame (29 bits identifier)	FALSE	Standard frame:
			TRUE	Extended Frame:
udiIDStart	UDINT	Start of the required area	eg. 000 0000 0010	
udiIDStop	UDINT	End of the required area	eg. 000 0000 1000	
* ... preset value				

### Output parameter

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

Parameter	Data type	Description	Possible values	
xError	BOOL	Indication if an error occurred during the FB execution	TRUE	<ul style="list-style-type: none"> <li>An error has 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  n	No CAN messages received between 2 FB calls  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          | Status: FB/Function is inactive.  |
| • STAT_DONE              | Status: 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<br>Contact the ifm Service Center!   |
| • ERR_UNDEFINED          | Error: Unknown error<br>Contact the ifm Service Center!   |

## 9.6.8 CAN\_RxRangeExt

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



### Description

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  $\leq$  the value of the upper limit.

#### Example:

Lower limit 0 0000 0000 0000 0000 0000 0010

Upper limit 0 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 ( $\rightarrow$  279)

FB for standard Identifier (11 bits): CAN\_RxRange ( $\rightarrow$  283)



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 parameter

Parameter	Data type	Description	Possible values
xEnable	BOOL	Control activity of the FB	<ul style="list-style-type: none"> <li>FALSE: Deactivate FB</li> <li>TRUE: Enable FB</li> </ul>
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	$\rightarrow$ CAN_CHANNEL (ENUM)
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 parameter

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>An error has 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): 0 ... 536.870.911</li> </ul>	

### Diagnostic codes:

- STAT\_INACTIVE      Status: FB/Function is inactive.
- STAT\_DONE      Status: 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!

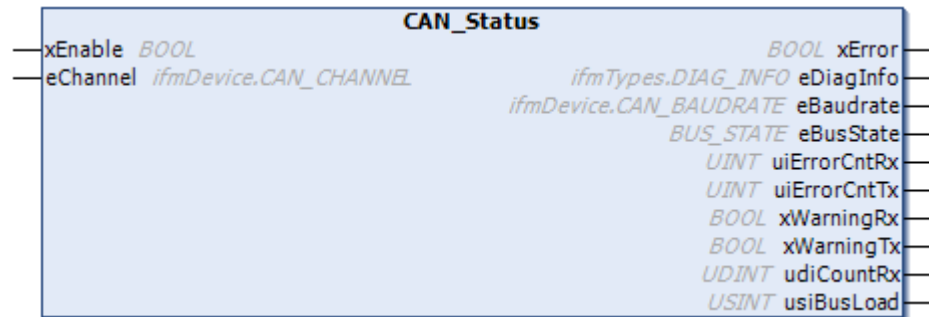
## 9.6.9 CAN\_Status

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



### Description

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

### Input parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ CAN_CHANNEL (ENUM)	

### Output parameter

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>• An error has 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	→ CAN_BAUDRATE (ENUM)	
eBusState	BUS_STATE	current state of the CAN interface	→ BUS_STATE (ENUM)	

Parameter	Data type	Description	Possible values	
uiErrorCntRx	UINT	Rx messages error counter	0...65535	
uiErrorCntTx	UINT	Tx messages error counter	0...65535	
xWarningRx	BOOL	Rx error: Threshold for warning messages is exceeded (uiErrorCntRx > 96)	FALSE	no warning
			TRUE	Warning
xWarningTx	BOOL	Tx error: Threshold for warning messages is 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!



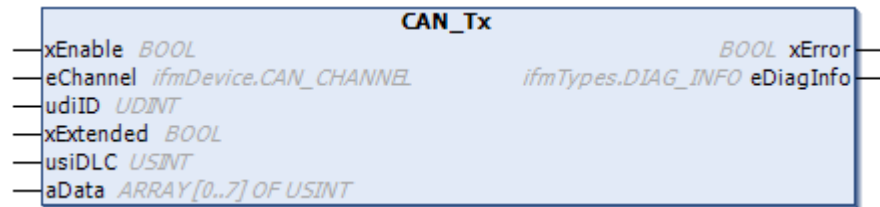
## 9.6.10 CAN\_Tx

**Function block type:** Function block (FB)

**Behaviour model:** ENABLE

**Library:** ifmRawCAN.library

**Symbol in CODESYS:**



### Description

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 parameter

Parameter	Data type	Description	Possible values	
xEnable	BOOL	Control activity of the FB	FALSE	Deactivate FB
			TRUE	Enable FB
eChannel	CAN_CHANNEL	Identifier of the CAN Interface	→ CAN_CHANNEL (ENUM)	
udiID	UDINT	Identifier of the CAN message	• For Standard Frame (11 bits identifier): 0 ... 2047 • for Extended-Frame (29 bits identifier): 0 ... 536.870.911	
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 parameter

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

Parameter	Data type	Description	Possible values
xError	BOOL	Indication if an error occurred during the FB execution	TRUE <ul style="list-style-type: none"> <li>An error has 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                      Status: FB/Function is inactive.
- STAT\_DONE                          Status: 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!

### 9.6.11 BUS\_STATE (ENUM)

Name	Description	Possible values		Data type	Value
BUS_STATE	State of the CAN interface	UNDEFINED	Interface not available or not configured	INT	0
		ERROR_ACTIVE	Error counter Tx and Rx <= 96	INT	1
		ERROR_PASSIVE	Error counter Tx or Rx > 127 and error counter Tx or Rx < 255	INT	2
		ERROR_WARNING	Error counter Rx or Tx > 96 and error counter Rx or Tx <= 127	INT	3
		BUS_OFF	Error counter Tx = 255	INT	65535

### 9.6.12 CAN\_Info (GVL)

Name	Description	Data type	Possible values	
CAN_State	State of the CAN channels	ARRAY[0...3] OF CAN_BUS_STATE	per array field: → CAN_BUS_STATE (STRUCT)	
Logger_Enabled	shows if the log of the CAN channel is active	ARRAY[0...3] OF BOOL	per array field:	
			FALSE	deactivated
			TRUE	activated

### 9.6.13 CAN\_BUS\_STATE (STRUCT)

Name	Data type	Description	Possible values	
uiBaudrate	UINT	Baud rate of the CAN interface	→ CAN_BAUDRATE (ENUM)	
eBusState	BUS_STATE	current state of the CAN interface	→ BUS_STATE (ENUM)	
uiErrorCntRx	UINT	Rx messages error counter	0...65535	
uiErrorCntTx	UINT	Tx messages error counter	0...65535	
xWarningRx	BOOL	Rx error: Threshold for warning messages is exceeded (uiErrorCntRx > 96)	FALSE	no warning
			TRUE	Warning
xWarningTx	BOOL	Tx error: Threshold for warning messages is exceeded (uiErrorCntTx > 96)	FALSE	no warning
			TRUE	Warning
udErrorCntTx	UDINT	Number of detected CAN messages (independent of configured Rx messages)	0...4294967295	

## 10 Appendix

### 10.1 Address assignment in Ethernet networks



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`, then  $256 \times 256 = 65\,024$  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 are possible
192.168.111.213	255.255.0.0	192.168.222.123	255.255.0.0	Yes, 65 024 participants are 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