SEW EURODRIVE

System Manual

PFF-HM31A Decentralized Safety Controller for **MOVIPRO®**

Edition 05/2012

19419619/EN


KKFU 144794 3
42G1

1 General Information

This system manual contains information about the designated use of the safety controller.

The following are required for safe installation, startup and safety during operation and maintenance:

- Knowledge of regulations
- Technically flawless implementation of the safety notes in this manual by qualified employees

Under the following circumstances, disruption or impairment of safety functions can cause severe injury to persons, damage to property or damage to the environment, for which SEW-EURODRIVE cannot assume liability:

- Unskilled access to the units
- Turning off or bypassing safety functions
- Not complying with the notes in this manual

SEW-EURODRIVE develops, manufactures and tests safety controllers while meeting applicable safety standards. The units may only be used if the following requirements have been met:

- Only the intended applications included in the descriptions
- Only the specified environmental conditions
- Only in connection with approved non-SEW units

1.1 Documentation design and use

This system manual covers the following topics:

- General Information
- System Properties
- Communication
- safeethernet
- Modbus TCP/UDP
- Com-user task (CUT)
- Operating System
- User Program
- Startup
- Operation
- Maintenance

This manual describes the following variant:

Programming tool	Processor operating system	Communications operating system
SILworX	CPU-BS V.8 or later	COM-BS V.13 or later

1.2 Target group

This document is intended for planners, project designers and programmers of automation systems, as well as persons who are entitled to start up, operate or maintain the units and systems. Specialized knowledge of safety-related automation systems is required.

1.3 Text conventions

For better legibility and for clarification, the following styles are used in this document:

Style	Meaning
Bold	Emphasizes important parts of the text.
[...]	Names of buttons and menu commands you can click on in the programming tool.
Italics	Parameters and system variables.
`Courier`	Actual user entries.
RUN	Names of operating states in capital letters.

1.4 Structure of the safety notes

1.4.1 Meaning of signal words

The following table shows the grading and meaning of the signal words for safety notes, warnings regarding potential risks of damage to property, and other notes.

Signal word	Meaning	Consequences if disregarded
⚠ **DANGER**	Imminent danger	Severe or fatal injuries
⚠ **WARNING**	Possible dangerous situation	Severe or fatal injuries
⚠ **CAUTION**	Possible dangerous situation	Minor injuries
IMPORTANT	Possible damage to property	Damage to the drive system or its environment
NOTE	Useful information or tip: Simplifies handling of the drive system.	

1.4.2 Structure of the section-related safety notes

Section-related safety notes do not apply to a specific action, but to several actions pertaining to one subject. The used symbols indicate either a general or a specific hazard.

This is the formal structure of a section-related safety note:

⚠ SIGNAL WORD

Nature and source of danger.

Possible consequence(s) if disregarded.

- Measure(s) to avoid the danger.

1.4.3 Structure of the embedded safety notes

Embedded safety notes are directly integrated in the instructions just before the description of the dangerous action.

This is the formal structure of an embedded safety note:

- ⚠ **SIGNAL WORD** Nature and source of danger.

  Possible consequence(s) if disregarded.

  – Measure(s) to avoid the danger.

1.5 Rights to claim under limited warranty

A requirement of fault-free operation and fulfillment of any rights to claim under limited warranty is that you adhere to the instructions in the documentation. Read the documentation before you start working with the unit.

1.6 Exclusion of liability

You must comply with the information contained in this documentation to ensure safe operation and to achieve the specified product characteristics and performance features. SEW-EURODRIVE assumes no liability for injury to persons or damage to equipment or property resulting from non-observance of these operating instructions. In such cases, any liability for defects is excluded.

1.7 Other applicable documentation

Observe the following applicable documents:

- "PFF-HM31A Decentralized Safety Controller for MOVIPRO®" operating instructions
- "PFF-HM31A Decentralized Safety Controller for MOVIPRO®" safety manual
- Drive Engineering - Practical Implementation, "Electromagnetic Compatibility (EMC) in Drive Engineering"

If you want to use the CUT function, please also observe the following applicable documents:

- "Com-User Task for PFF-HM31A" manual
- "MOVIVISION® Configuration and Diagnostics Tool Version 2.0" manual

You require software which is **not** included in the scope of delivery. You can obtain the software and documentation from SEW-EURODRIVE on CD or DVD by providing the following order information:

Designation	Item number
SILWorX for PFF-HM31A • Hardware: SILWorX license dongle • Software: SILWorX 4.64.0 or later	1 950 011 4
Motion Library PFF-HM31 Function block library for safety-related position detection	1 710 640 0

Also note the applicable documents depending on the drive technology connected.

The current version of the documentation/software can be downloaded from the SEW-EURODRIVE homepage (www.sew-eurodrive.com) under "Documentation".

1.8 Copyright

1.9 Product names and brands

The product names mentioned in this documentation are brands or registered brands of the titleholders.

2 System Properties

The safety controller includes a safety-related processor system, a number of inputs and outputs, as well as communications connections, in a single housing unit.

Details are available in the "PFF-HM31A Decentralized Safety Controller for MOVIPRO®" operating instructions.

2.1 Monitoring the operating voltage

The unit monitors the 24 V DC voltage during operation. It reacts based on the listed voltage levels:

Voltage level	Reaction of the units
DC 24 V -20%/+25% (19.2 V to 30 V)	Normal operation
<18.0 V (circuit board voltage read by the software)	Alarm status (internal variables are described and sent to inputs and outputs)
<12.0 V (circuit board voltage read by the software)	Inputs and outputs are turned off

The power supply status system variable enables you to analyze the operating voltage status with the programming tool or in the user program.

2.2 Monitoring the temperature status

The temperature is measured by sensors at relevant locations inside the unit or system and is output by the software.

It has a delta amount relative to the ambient temperature; this amount depends on many factors. If defined circuit board temperatures (two switching points) are reached, the safety controller switches to the safe status.

If the temperature measured inside the unit exceeds the defined switching points, the value of the temperature state system variable changes as follows:

Temperature (inside the unit)	Temperature range	Temperature status [BYTE]
<60°C	Normal	0x00
60° to 70°C	High temperature	0x01
>70°C	Very high temperature	0x03
Return to 64°C to 54°C[1)]	High temperature	0x01
Return to <54°C[1)]	Normal	0x00

1) The sensors have a hysteresis of 6°C.

If there is little or no air circulation and natural convection is insufficient, the threshold for the "high temperature" range in the safety controller can be tripped at ambient temperatures of <35°C. This could be caused be local warming or poor heat conduction. Especially for digital outputs, warming depends to a large extent on the load. The temperature status system variable allows the user to read the internal temperature.

NOTE

The transition to "high temperature" or "very high temperature" status does not mean that system safety has been compromised.

2.3 Alarm and event recording

The safety controller can record alarms and events (sequence of events recording, SER).

2.3.1 Alarms and events

Events are changes in the status of the plant or controller which are marked with a time stamp.

Alarms are events that signal an increased risk potential.

The safety controller records changes in status as events, including the time when they occurred.

The safety controller distinguishes between Boolean and scalar events.

Boolean events:

- Changes in Boolean variables, e.g. of digital inputs.
- Alarm and normal status; these statuses can be freely assigned to the variable statuses

Scalar events:

- Exceeding of limits that have been defined for a scalar variable.
- Scalar variables have a numerical data type, e.g. INT, REAL.
- Two upper and two lower limits can be set.
- The following must apply to the limits:

  Maximum limit = upper limit = normal range = lower limit = lowest limit.
- A hysteresis can take effect in the following cases:
  - If a value falls below an upper limit.
  - If a value exceeds a lower limit.

Entering a hysteresis prevents an unnecessarily large number of events if the global variable fluctuates a great deal around a limit.

The safety controller can only create events if the events have been defined in SILworX; see the "Configuring alarms and events" section. Up to 4,000 alarms and events can be defined.

2.3.2 Creating events

The processor system can create events. The processor system creates the events from global variables and stores them in the buffer; see "Event recording". Events are created in the user program cycle. Each event that has been read can be overwritten by a new event.

System events:

In addition to events that register changes in global variables or input signals, the processor systems create the following types of system events:

- Overflow: Events have not been stored because of a buffer overflow. The timestamp of the overflow event is the time when the event that caused the overflow occurred.
- Init: The event buffer has been initialized.

System events include the SRS identification of the unit that caused the event. Status variables make the event status of scalar events available to the user program. A BOOL-type global variable can be assigned to each of the following statuses as a status variable:

- Normal.
- Fell below lower limit.
- Fell below lowest limit. Upper limit exceeded.
- Maximum limit exceeded.

The assigned status variable becomes TRUE if the corresponding status is reached.

2.3.3 Event recording

The processor system collects the events: The processor system stores all the events in its buffer. The buffer is located in the non-volatile memory and holds 1,000 events. If the buffer is full, no new events are saved until more events have been read, and as a result marked for overwriting.

2.3.4 Transferring events

The events can be transferred to the drive control (Beck PC) via the MODBUS protocol, or to the higher-level safety controller via safeethernet. For this purpose, the corresponding variables must have been previously linked in the user program. Advanced diagnostics is performed via the PADT (SILworX).

3 Communication

The safety controllers communicate using the following protocols:

- safeethernet

  Safety-related protocol for controllers communicating with each other
- Modbus TCP/UDP fieldbus protocol for connecting external units or systems
- Communication with the programming unit

The communications system is connected to the safety-related processor system.

It and the fieldbus interfaces are connected to the secure microprocessor system via dual-port RAM. Only units that ensure safe electrical separation may be connected to the interfaces.

The communications system controls the controller's communication with other systems via high-performance interfaces:

Available protocols

The following protocols are available:

Protocol	Interfaces	Activation
safeethernet	Ethernet	The function is enabled by default for the unit option PFF-HM31A1-E61-I111-00/000/000.
SNTP server/client	Ethernet	
Modbus TCP master	Ethernet	
Com-user task	CAN (X4111_1/2) RS485 (X4011)	

Optional protocols

The following protocols are available as unit options upon request:

Protocol	Interfaces	Activation
Modbus TCP slave	Ethernet	A new unit option with the desired protocol enabled can be generated upon request.
TCP send/receive		
PROFINET IO Controller		
PROFINET IO Device		
OPC server (runs on host PC)		

NOTE

In unit option PFF-HM31A1-E61-I111-00/000/000, the optional protocols can be used for testing purposes without activation for 5,000 operating hours. The "ERROR" system LED lights up red continuously when protocols that were not enabled are used.

After the 5,000 operating hours have elapsed, the controller no longer starts.

- Order the unit option with the required protocols in a timely manner.

3.1 Ethernet

The safety controller features an Ethernet switch with connections. Other units can be connected to the controller via these connections using Ethernet cables.

The following interfaces are available:

- **2 Ethernet interfaces:** X4233_1 and X4233_2

  Both interfaces are located on the unit's terminal strip
- **1 Ethernet service interface:** X4223

  For connecting a programming unit (PADT)

Switch

- Unlike a hub, a switch can analyze data packets and store them temporarily. The switch then creates a temporary specific connection between two communication partners (sender/recipient) to transfer the data. This avoids the collisions that often occur with hubs, and reduces the network load. Each switch requires an address/port assignment table for directed data transfer. This table is automatically generated by the switch in a self-learning process. In the table, MAC addresses are assigned to a specific port on the switch. Incoming data packets are transferred directly to the corresponding port based on this table.
- The switch automatically moves between 10 and 100 Mbit/s transfer rates, as well as between full and half-duplex connections. This means the full bandwidth is available to each data transfer direction (full duplex operation).
- A switch controls communication between different communication terminals. It can address up to 1,000 absolute MAC addresses.
- Autocrossing detects the connection of cables with crossed wires and the switch automatically adjusts to them.

NOTE

The notes in the safety manual must be observed during configuration of safety-related communication.

3.1.1 SNTP protocol

With the SNTP protocol (Simple Network Time Protocol), the SNTP server synchronizes the time on the SNTP clients via Ethernet. The safety controller can be configured and used as an SNTP server and/or SNTP client.

The RFC 2030 SNTP standard (SNTP version 4) applies, with the limitation that only Unicast mode is supported.

- The function is enabled by default.
- Ethernet 10/100-BASE-T is required as the network transmission standard.

SNTP client

The SNTP client uses only the accessible SNTP server with the highest priority for time synchronization.

One SNTP client in each resource can be configured for time synchronization.

NOTE

Time synchronization of one safety controller by another safety controller.

If an SNTP client is set up on a safety controller, the safety controller's internal SNTP server is turned off.

To ensure continued time synchronization of one safety controller by another safety controller, an SNTP server must be set up in the communications module that is connected to the remote I/O.

This is how you set up a new SNTP client:

1. Open [Configuration]/[Resource]/[Protocols] in the structure tree.
2. Right-click on "Protocols" and select [New]/[SNTP client] in the context menu.
   A new SNTP client is added.
3. Select the COM module in the context menu of [SNTP client]/[Properties].

The SNTP client's dialog window contains the following parameters:

Parameter	Description
Type	SNTP client
Name	Name for the SNTP client, 32 characters max.
Module	Selecting the CPU or COM module where this protocol is processed.
Behavior if CPU/COM connection is lost	If the connection between the processor module and the communications module is lost, the input variables are either initialized or used unchanged in the processor module, based on this parameter. (For example, if the communications module is disconnected while communication is in progress.) **Accept initial data:** Input variables are reset to the initial values. **Maintain last value**: Input variables retain the last value.
Enable max. µP budget	Is disregarded by the module's operating system. Parameter was maintained for CRC and reload stability.
Max. µP budget in [%]	Is disregarded by the module's operating system. Parameter was maintained for CRC and reload stability.
Description	Any unique description of the SNTP
Current SNTP version	Displays the current SNTP version.
Reference stratum	An SNTP client's stratum indicates how precise the client's local time is. The lower the stratum, the more precise its local time. Zero means an unspecified or unavailable stratum (not valid). An SNTP client's currently used SNTP server is the server which is accessible and has the highest priority. If the stratum of the current SNTP server is smaller than that of the SNTP client, the resource applies the current SNTP server's time. If the stratum of the current SNTP server is larger than that of the SNTP client, the resource does not apply the current SNTP server's time. If the stratum of the current SNTP server is the same as that of the SNTP client, there are two different possibilities: • If the SNTP client (resource) only functions as an SNTP client, the resource applies the current SNTP server's time. • If the SNTP client (resource) also functions as an SNTP server, half of the time difference between it and the current SNTP server is applied to the resource for each SNTP client query (time adjusts gradually). Value range: 16 s to 16384 s (default value: 16 s)
Client time query interval [s]	Time interval during which the current SNTP server synchronizes time. The client time query interval in the SNTP client must be larger than the timeout in the SNTP server. Value range: 16 s to 16384 s (default value: 16 s)

SNTP client (server info)

The connection to an SNTP server is configured in the SNTP server info. One to four SNTP server infos can be configured under an SNTP client.

This is how you add a new SNTP server info:

1. Open [Configuration]/[Resource]/[Protocols]/[SNTP client] in the structure tree.
2. Right-click on "Protocols" and select [New]/[SNTP server info] in the context menu.
   A new SNTP server info is added.
3. Select the COM module in the context menu of [SNTP server info]/[Properties].

The SNTP server info dialog window contains the following parameters:

Parameter	Description
Type	SNTP server info
Name	Name of the SNTP server info. 31 characters max.
Description	Description of the SNTP server. 31 characters max.
IP address	IP address of the resource or the PC on which the SNTP server is configured. Default value: 0.0.0.0
SNTP server priority	Priority with which the SNTP client handles this SNTP server. SNTP servers configured for a single SNTP client should have different priorities. Value range: 0 (lowest priority) through 4294967295 (highest priority) Default value: 1 s
SNTP server timeout [s]	The timeout in the SNTP server must be set lower than the time query interval in the SNTP client. Value range: 1 s to 16384 s Default value: 1 s

SNTP server

The SNTP server accepts the query from the SNTP client and sends its current time back to the SNTP client.

This is how you add a new SNTP server:

1. Open [Configuration]/[Resource]/[Protocols] in the structure tree.
2. Right-click on "Protocols" and select [New]/[SNTP server] in the context menu.

   A new SNTP server is added.
3. Select the COM module in the context menu of [SNTP server]/[Properties].

The SNTP client dialog window contains the following parameters:

Parameter	Description
Type	SNTP client
Name	Name for the SNTP client, 31 characters max.
Module	Selecting the CPU or COM module where this protocol is processed.
Enable max. µP budget	**Activated:** Copy the µP budget limit from the "Max. µP budget in [%]" field. **Deactivated:** Do not use a µP budget limit for this protocol.
Max. µP budget in [%]	Maximum µP load for the module which can be produced while processing the protocol. Value range: 1 to 100% Default value: 30%
Behavior if CPU/COM connection is lost	If the connection between the processor module and the communications module is lost, the input variables are either initialized or used unchanged in the processor module, based on this parameter. (For example, if the communications module is disconnected while communication is in progress.) **Accept initial data:** Input variables are reset to the initial values. **Maintain last value**: Input variables retain the last value.
Description	Any unique description of the SNTP
Current SNTP version	Displays the current SNTP version.
Time server stratum	An SNTP client's stratum indicates how precise the client's local time is. The lower the stratum, the more precise its local time. Zero means an unspecified or unavailable stratum (not valid). The SNTP server stratum must be lower than or the same as the stratum of the querying SNTP client. Otherwise the SNTP server's time will not be applied to the SNTP client. Value range: 1 to 15 Default value: 14

3.2 Communication with the programming tool

The safety controller communicates with a PADT via Ethernet. A PADT is a PC/laptop on which the SILworX programming tool has been installed.

The controller can communicate with up to 5 PADTs at a time. However, only one programming tool can write to the controller. All other PADTs can only read information. The controller provides read-only access to any other PADT attempting to write.

4 safeethernet

The safety controller is safeethernet-enabled. It can communicate via Ethernet (100 Mbit/s) in safety-related mode according to SIL 3.

The safety controller's Ethernet interfaces can also be used simultaneously for other protocols.

The controllers can communicate via safeethernet using different Ethernet network topologies. Adjust the safeethernet protocol parameters to the Ethernet network used in order to increase the data transfer speed and efficiency.

These parameters can be set using what are known as network profiles. The parameters' factory configuration ensures communication so that the user does not have to learn network configuration details right away.

NOTE

The safeethernet protocol is safety-related and certified by TÜV [German Technical Control Board] up to SIL 3 according to IEC 61508.

safeethernet properties

Element	Properties	Description
Required module/controller	Controller's integrated processor module	safeethernet is run on the safety-related processor module.
Ethernet interfaces	100 Mbit/s	The Ethernet interfaces used can also be used simultaneously for other protocols.
Connections	128	safeethernet connections
Redundant connections:	128	2-channel operation of redundant safeethernet connections between controllers can be set in the safeethernet editor.
Redundant transmission routes	Limitation because only one unit	Redundant safeethernet transmission routes
Amount of process data per connection	1100 bytes	per safeethernet connection

4.1 What is safeethernet?

Requirements such as determinism, reliability, interchangeability, expandability and, above all, safety, are key issues in the process and automation technology sector.

safeethernet is a transfer protocol for transferring safety-related data up to SIL 3 using Ethernet technology.

safeethernet includes mechanisms that recognize the following errors and cause a safety-related response to them:

- Corruption of transmitted data (duplicate, lost, modified bits)
- Incorrect message addressing (sender, recipient)
- Incorrect data sequence (repetition, loss, switch)
- Incorrect time behavior (delay, echo)

safeethernet is based on the IEEE 802.3 standard.

safeethernet uses "insecure data transfer channels" (Ethernet) in accordance with the black channel principle, and monitors them at the sender's and recipient's ends through safety-related protocol mechanisms. This allows the use of Ethernet network components such as hubs, switches and routers within a safety-related network.

safeethernet uses the functions of standard Ethernet in a way that enables safety and real-time capability. A special protocol mechanism guarantees deterministic behavior even when communication stations are dropped or enter the network. The system then automatically integrates new components while it is running. All network components can be exchanged during live operation. Transmission times can be clearly defined using switches. This enables Ethernet's real-time capability.

Connections to the company's internal Intranet, as well as connections to the Internet, are possible with safeethernet. This way, only one network is required for both secure and non-secure data transfers.

NOTE

The network can also be used by other participants if sufficient transmission capacity is available.

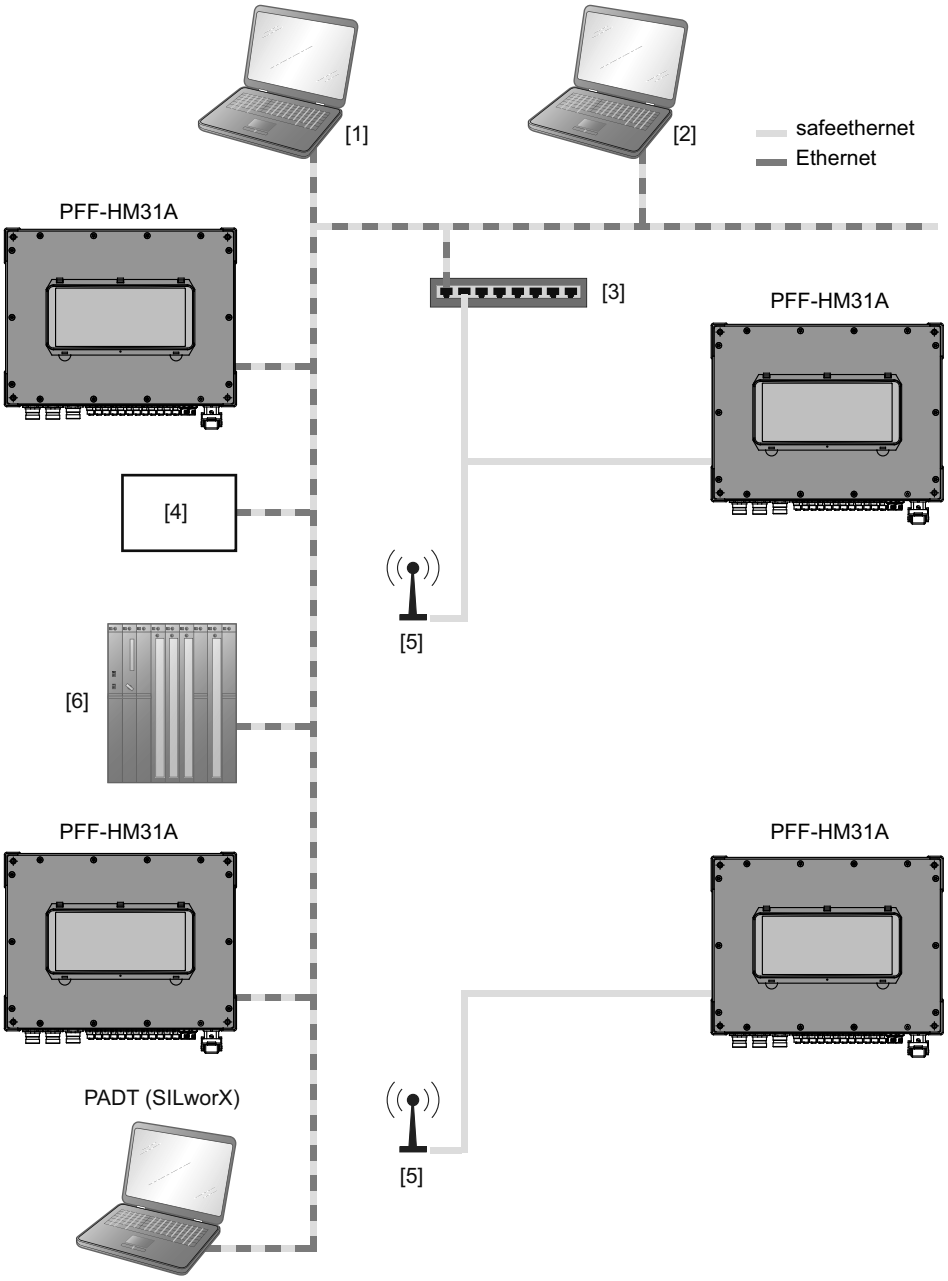
WARNING

Manipulation of safety-related data transfer

Severe or fatal injuries.

The operator must ensure that the Ethernet used for safeethernet is sufficiently protected from manipulation (e.g. by hackers). The type and extent of the measures taken must be coordinated with the relevant inspection authorities.

safeethernet enables flexible system structures for decentralized automation with specified response times. Depending on your requirements, you can choose to have central or decentralized distribution of intelligence to participants within the network.



5519919883

[1] PC for the DCS guiding system
[2] PADT (SILworX)
[3] Switch
[4] DCS guiding system
[5] Radio, satellite, WLAN, fiber optics, ISDN or DSL
[6] PLC

NOTE

An unintentional change to safe status is possible

- Take care that no network loops are created during interconnection. Data packets can only reach the controller on **one** path.
- Only use managed switches when setting up an Ethernet ring topology.

4.2 safeethernet editor

You can create and configure the safeethernet connections to communication partners (resources) in the safeethernet editor.

This is how you open the local resource's safeethernet editor:

1. Open [Configuration]/[Resource] in the structure tree.
2. Right-click on "safeethernet" and select [Edit] in the context menu.

The safeethernet editor contains the work area and the selection of objects.

You can create and configure the safeethernet connections to communication partners (resources) in the safeethernet editor. To do so, drag the resources from the selection of objects into the work area.

To configure the safeethernet connection, you must set the following safeethernet protocol parameters:

Parameter	Description
Partner	Resource name of the link partner
IF CH...	Available Ethernet interfaces on the resource (local) and the resource (target)
Profile	Combination of matching safeethernet parameters; see also the "safe-ethernet profiles" section
Response time [ms]	Time until receipt of a message is acknowledged at the sender; see also the "Response time" section.
Receive timeout [ms]	Monitoring time on PES1 during which a correct answer must be received from PES2; see also the "Receive timeout" section
Resend timeout [ms]	Monitoring time on PES1 during which PES2 must acknowledge receipt of a data packet. Otherwise the data packet is resent; see also the "Resend timeout" section.
Acknowledge timeout [ms]	Interval during which a received data packet must be acknowledged by the CPU; see also the "Acknowledge timeout" section.
Prod. rate	Production rate: Smallest interval between two data packets; see also the "Production rate" section.
Memory (queue depth)	Number of data packets that can be sent without acknowledgement; see also the "Memory" section.
Freeze data in case of dropped connection [ms]	Behavior of this safeethernet connection's input variables if the connection is interrupted.[1)] • Use initial data: The initial data is used for the input variables. Unlimited Input variables are frozen at their current values and used until the connection is reestablished. • Limited Entry: Double-click the drop-down menu and enter the time. The input variables are frozen at their current values and used until after the timeout parameter set. Afterwards the initial data is used. The timeout can be extended by up to one CPU cycle.
Fragments per cycle	Fixed setting: One fragment per controller cycle is transmitted to the communication partner. Fragment ≤ 900 bytes
Event priority	Function is not supported.
Priority of status values	
Number of ignored warnings	This is the number of warnings which must occur consecutively during the "Warning period [ms]" time span before the warnings are entered into diagnostics or the communications error statistics.
Warning period [ms]	0 ms is currently the only allowable value.
Activate SER	Default value: Deactivated

1) Observe this warning note:

⚠ WARNING

Behavior of input variables if connection is interrupted

Severe or fatal injuries.

Only one "Use initial data" setting can be applied for safety-related functions which are implemented via safeethernet.

Object selection

Object selection provides all resources within this project which the resource can connect to via safeethernet.

4.3 Detail view of the safeethernet editor

The detail view always refers to the local resource for which you started the safeethernet editor.

This is how you open the detail view for a safeethernet connection:

1. Open the context menu by right-clicking on [safeethernet connection].
2. Click on [Detail view].

   The detail view includes the system variables register, fragment definitions and Resource (local)<->Resource (target).

4.3.1 Register: System variables

You can control the safeethernet connection and analyze its status using system variables in the user program.

System variable	Description	
Ack. frame no.	Receipt counter (revolving).	
Number of faulty messages	Number of all faulty messages per channel (incorrect CRC, incorrect header, other errors)	
Number of faulty messages in the red. channel		
Number of successful connections	Number of successful connections since the statistics were reset.	
Number of lost messages	Number of messages lost on one of the two transmission routes since the statistics were reset. The counter is only run until a channel fails completely.	
Number of lost messages in the red. channel		
Early queue usage	Number of messages which were placed into the early queue since the statistics were reset; see also the "Memory" section.	
Faulty messages	Number of messages rejected since the statistics were reset.	
Frame no.	Sent counter (revolving)	
Channel status	Current channel status of channel 1. The channel status is the current status of channel 1 at the time (seq. no. X-1) a message with seq. no. X was received.	
	Status	**Description**
	0	No message regarding the status of channel 1.
	1	Channel 1 OK.
	2	Last message was faulty; current one is OK.
	3	Error on channel 1.
Layout version	Signature of the data layout used in the communication.	

System variable	Description		
Last channel latency	The channel latency specifies the delay between the two redundant transmission paths at the time when messages with identical seq. nos. were received. Statistics of the average, minimum, maximum and most recent latencies are kept for this purpose. If the min. value > the max. value, the statistical values are not valid. The most recent channel latency and the average channel latency are then 0.		
Most recent latency of the red. channel			
Max. channel latency			
Max. channel latency of the red. channel			
Min. channel latency			
Min. channel latency of the red. channel			
Average channel latency			
Average channel latency of the red. channel			
Monotonicity	User data sent counter (revolving).		
New layout version	Signature of the new data layout.		
Quality of channel 1	Status of the main transmission route.		
	Bit no.	**Bit = 0**	**Bit = 1**
	0	Transmission route not activated	Transmission route activated
	1	Transmission route not used	Transmission route used actively
	2	Transmission route not connected	Transmission route connected
	3	-	Transmission route delivers message first
	4–7	Reserved	Reserved
Quality of channel 2	Status of the redundant transmission route; see status of channel 1 (main transmission route).		
Receive timeout	Time in milliseconds (ms) on PES1 during which a valid answer from PES2 must be received; see also the "Receive timeout" section		
Response time	Time in milliseconds (ms) until receipt of a message is acknowledged at the sender; see also the "Response time" section		
Reset safeethernet statistics	Reset statistical values for the communications connection in the user program (e.g. number of faulty messages, channel status, timestamp of the most recent error in the red. channel, etc., retries).		
	Value	**Function**	
	0	No reset	
	1–255	Reset of the safeethernet statistics	
Transmission control channel1	Transmission control of channel1		
	Bit 0	**Function**	
	FALSE	Transmission route activated for tests	
	TRUE	Transmission route blocked	
	Bits 2 through 7 reserved.		
Transmission control channel2	See transmission control channel 1.		
Connection control	The user program can control the safeethernet connection using these system variables.		
	Command	**Description**	
	Autoconnect (0x0000)	Default value: After safeethernet communication has been lost, the controller attempts to reestablish connection during the next CPU cycle.	
	Toggle mode 0 (0x0100) Toggle mode 1 (0x0101)	After communication has been lost, a program-controlled change in the toggle mode can reestablish the connection. • TOGGLE MODE_0 (0x100) set: Set to TOGGLE MODE 1 (0x101) to reestablish the connection. • TOGGLE MODE 1 (0x101) set: Set to TOGGLE_MODE_0 (0x100) to reestablish the connection.	
	Disabled (0x8000)	safeethernet communication turned off.	

System variable	Description	
Connection status	The connection status analyzes the status of the communication between two controllers in the user program.	
	Status/value	**Description**
	Closed (0)	Connection is closed and no attempt is being made to open it.
	Try_open (1)	An attempt is being made to open the connection, but it is not yet open. This status applies to both the active and the passive side.
	Connected (2)	The connection has been established and is in operation (active time monitoring and data exchange)
Retries	Number of retries since the statistics were reset.	
Timestamp of the last error in the red. channel [ms]	Millisecond portion of the timestamp (current system time).	
Timestamp of the last error in the red. channel [s]	Seconds portion of the timestamp (current system time).	
Timestamp of the last error [ms]	Millisecond portion of the timestamp (current system time).	
Timestamp of the last error [s]	Seconds portion of the timestamp (current system time).	
Status of the red. channel	Current channel status of channel 2. The channel status is the current status of channel 2 at the time (seq. no. X-1) a message with seq. no. X was received.	
	Status	**Description**
	0	No message regarding the status of channel 2
	1	Channel 2 OK
	2	Last message was faulty; current one is OK.
	3	Error on channel 2.

4.4 safeethernet parameters

You can set up safety-related communication in the safeethernet editor. To do so, you must set the parameters described in this section. The following condition applies to calculating the *Receive Timeout* and *Response Time* safeethernet parameters: The communications time slice must be large enough to process all safeethernet connections during one CPU cycle; see the "Maximum communications slice" section.

4.4.1 Maximum cycle time of the safety controller

SEW-EURODRIVE recommends the following method for determining the maximum cycle time of a PFF-HM31A safety controller:

Determining maximum cycle time of the PFF-HM31A safety controller:

1. Operate system at full load. In the process, all communications connections must be in operation, both via safe**ethernet** and via standard protocols. Read the cycle time in the control panel frequently and note the maximum cycle time.
2. Repeat step 1 for the communication partner (second safety controller).
3. The larger of the two cycle times determined is the desired maximum cycle time.

The maximum cycle time has been determined and is used in the calculations below.

4.4.2 Receive timeout

ReceiveTMO is the monitoring time in milliseconds (ms) during which a correct answer must be received from the communication partner.

If no correct answer has been received from the communication partner during the *ReceiveTMO*, the safety-related communication is closed. The input variables of this safe**ethernet** connection behave according to the *Freeze data if connection is lost [ms]* parameter set.

For safety-related functions which are implemented via safe**ethernet,** only the *Use initial data* setting may be used.

Because the *ReceiveTMO* is safety-related and forms part of the Worst Case Reaction Time T_R (maximum reaction time; see safety manual, Section 8.2.4), the *ReceiveTMO* must be calculated as follows and entered into the safe**ethernet** editor:

ReceiveTMO ≥ 4 × delay + 5 × max. cycle time

Condition: The communications time slice must be large enough to process all safe **ethernet** connections during one CPU cycle.

Delay: Delay on the transmission path, e.g. due to switch, satellite

Max. cycle time: Maximum cycle time of both controllers

NOTES

- A desired communications error tolerance can be achieved by increasing the *Receive TMO*, if such timing is permitted for this application process.
- The maximum allowable value for *Receive TMO* depends on the application process and is set in the safe**ethernet** editor along with the maximum expected *Response Time* and the profile.

4.4.3 Response time

The *ResponseTime* is the time in milliseconds (ms) that elapses until the sender of a message receives an acknowledgement from the recipient.

To set parameters using a safe**ethernet** profile, an expected *ResponseTime* must be specified based on the physical conditions of the transmission path.

The specified *ResponseTime* affects the configuration of all parameters for the safe **ethernet** connection; these parameters must be calculated as follows:

ResponseTime ≤ ReceiveTMO / n

n = 2, 3, 4, 5, 6, 7, 8, etc.

The ratio of *ReceiveTMO* to *ResponseTime* affects the error tolerance capability, e.g. if packets are lost (retrying lost data packets) or there are delays on the transmission route.

In a network where packets could be lost, the following condition must be met:

Min. Response Time ≤ ReceiveTMO / 2 ≥ 2 × delay + 2.5 × max. cycle time

If that condition has been met, the loss of at least one data packet can be handled without the safeethernet connection being interrupted.

NOTES

- If this condition has not been met, the availability of a safeethernet connection can only be guaranteed in a network free of collisions and interference. This, however, does not mean that the processor module has a safety problem.
- It must be ensured that the communications system complies with the *ResponseTime* parameter set.

  In cases where this cannot always be guaranteed, a corresponding system variable for the connection is available to monitor the *ResponseTime*. If the *ResponseTime* measured is exceeded by half of the *ReceiveTMO* more often than in exceptional cases, the *ResponseTime* set must be increased.

  The *ReceiveTimeout* must be adjusted to the new *ResponseTime* set.
- In the examples below, the formulas for calculating the maximum response time for a connection with the safety controller only apply if the safety time for these has been set to = 2x the watchdog time.

4.4.4 Sync/Async

Sync: Currently not supported.

Async: Is the default setting. When the parameter is set to Async, the safeethernet protocol instance receives during the CPU's input phase and sends according to its sending rules during the CPU's output phase.

4.4.5 ResendTMO

ResendTMO cannot be entered manually, but is calculated from the profile and the *ResponseTime*. Monitoring time in milliseconds (ms) on PES 1 during which PES 2 must acknowledge receipt of a data packet; otherwise the data packet is retried.

Rule: *ResendTMO* ≤ *ReceiveTimeout*

If resend timeouts have been configured differently for the communication partners, the active protocol partner (lower SRS) determines the actual value of the protocol connection's resend timeout.

4.4.6 Acknowledge timeout

AckTMO cannot be entered manually, but is calculated from the profile and the *ResponseTime*. *AckTMO* is the latest time after which a received data packet must be acknowledged by the CPU.

In a fast network *AckTMO* is zero, i.e., the receipt of a data packet is acknowledged immediately. In a slow network (e.g. a telephone modem path), *AckTMO* is greater than zero. In this case, the system tries to transmit the acknowledgement along with the process data to reduce the network load by avoiding addressing and safety blocks.

Rules:

AckTMO must be ≤ *ReceiveTimeout*

AckTMO must be ≤ *ResendTimeout*, if *ProductionRate* is > *ResendTimeout*.

4.4.7 Production rate

ProdRate cannot be entered manually, but is calculated from the profile and the *ResponseTime*.

Smallest interval in milliseconds (ms) between two data packets.

The goal of ProdRate is to limit the number of data packets to a quantity that does not overload a (slow) communications channel. This achieves a consistent utilization of the transmission medium and prevents receipt of old data at the receiving end.

Rules:

- ProdRate ≤ *ReceiveTimeout*
- ProdRate ≤ *ResendTimeout*, if *AcknowledgeTimeout* > *ResendTimeout*

NOTE

A production rate of zero means that data packets can be transmitted in each cycle of the user program.

4.4.8 Memory

Memory cannot be entered manually, but is calculated from the profile and the *ResponseTime*.

Memory (queue depth) is the number of data packets that can be sent without having to wait for acknowledgement.

The value depends on the network's transmission capacity and possible delays due to network runtimes.

The available message memory in the CPU is divided among all safeethernet connections.

4.5 Maximum response time for safeethernet

In the examples below, the formulas for calculating the maximum response time only apply if the safety time has been set to = 2x watchdog time.

NOTE

The maximum allowable response time depends on the process and must be coordinated with the relevant inspection authorities.

Terms	Meaning
ReceiveTMO	Monitoring time in PES 1 during which a valid answer must be received from PES 2. Otherwise, the safety-related communication is closed after that time has elapsed.
Production rate	Minimum interval between two data transfers.
Watchdog time	Maximum allowable duration of a RUN cycle in a controller. The duration of the RUN cycle depends on the complexity of the user program and the number of safeethernet connections. Watchdog time (WDT) must be entered in the properties of the resource.
Worst case reaction time	Maximum reaction time for transmitting the signal change of a PES 1's physical input (In) until the change of a PES 2's physical output (Out).
Delay	Delay in a transmission path, e.g. for a modem or satellite connection. For a direct connection, a delay of 2 ms can initially be assumed. The actual delay on a transmission path can be measured by the network administrator in charge.

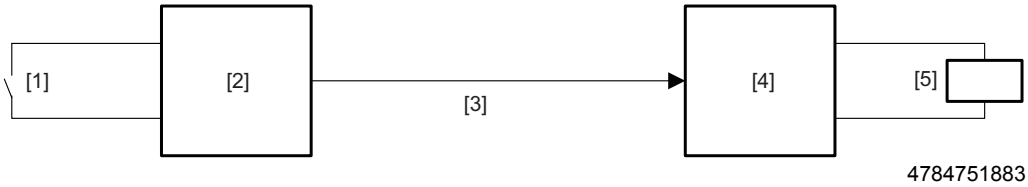
The following conditions apply to the calculation of maximum allowable response times below:

- The signals which are transmitted via safeethernet must be processed in each controller within one CPU cycle.
- The response times for the sensors and actuators must also be added.

The calculations also apply to signals in the opposite direction.

4.5.1 Calculating the maximum response time

The maximum response time T_R (worst case) from the change of an input in PES 1 until the response of the output in PES 2 can be calculated as follows:



4784751883

[1] Input
[2] PES 1 safety controller
[3] Safety-related protocol
[4] PES 2 safety controller
[5] Output

$T_R = t_1 + t_2 + t_3$

T_R Worst case reaction time
t_1 2 × watchdog time of safety controller 1
t_2 ReceiveTMO
t_3 2 × watchdog time of safety controller 2

Maximum response time depends on the process and must be coordinated with the relevant inspection authorities.

4.5.2 safeethernet profiles

safeethernet profiles are combinations of matching parameters which are set automatically when a safeethernet profile is selected. To set the parameters, only the receive timeout and the expected response time must be configured individually.

The goal of a safeethernet profile is to optimize data throughput in the network while taking into account physical conditions.

The following conditions are required for effective optimization:

- The communications time slice must be large enough to process all safeethernet connections during one CPU cycle.
- Mean CPU cycle time < response time.
- Mean CPU cycle time < ProdRate or ProdRate = 0

IMPORTANT

Improper combinations of CPU cycle, communications time slice, response time and ProdRate are not rejected during code generation and downloading/reloading. However, these combinations can cause faults, including failure of the safeethernet communication.

Possible damage to the drive system.

- Check the "**Faulty messages**" and "**Retries**" displays in the control panels of both controllers.

The safeethernet profile appropriate for this transmission path can be selected from the six available safeethernet profiles. Note the following warning:

⚠ WARNING

Only the **noisy profiles** are suitable for safety-related process data communication.

Severe or fatal injuries.

Only use the noisy profiles for safety-related process data communication:

- Fast&Noisy, Medium&Noisy and Slow&Noisy

The following table shows you the available profiles:

Profile	Application
Fast & Cleanroom	Only recommended for interference-free networks.
Fast & Noisy	Recommended for high availability of the safeethernet connection.
Medium & Cleanroom	Only recommended for interference-free networks.
Medium & Noisy	Recommended for high availability of the safeethernet connection.
Slow & Cleanroom	Only recommended for interference-free networks.
Slow & Noisy	Recommended for high availability of the safeethernet connection.

4.5.3 Profile I (Fast & Cleanroom)

⚠ WARNING

Only the **noisy profiles** are suitable for safety-related process data communication.

Severe or fatal injuries.

Only use the noisy profiles for safety-related process data communication:

- Fast&Noisy, Medium&Noisy and Slow&Noisy

Application

The **Fast & Cleanroom** profile is appropriate for applications in an ideal environment, e.g. a laboratory.

- For the fastest data throughput
- For applications that require fast data transmission
- For applications that require the lowest possible worst case reaction time

Network requirements
- Fast: 100 Mbit technology (100 BASE-TX), 1 Gbit technology
- Clean: Interference-free network.
- Data losses due to network overload, outside influences and network manipulation must be avoided.
- LAN switches required.

Communication path characteristics
- Minimal delays
- Expected response time ≤ ReceiveTMO

  (Otherwise ERROR when parameters are set)

4.5.4 Profile II (Fast & Noisy)

Application

The **Fast & Noisy** profile is the SILworX default profile for communication via safeethernet.
- For the fastest data throughput
- For applications that require fast data transmission
- For applications that require the lowest possible worst case reaction time

Network requirements
- Fast: 100 Mbit technology (100 BASE-TX), 1 Gbit technology
- Noisy: Network is not interference-free. Low likelihood of data packet loss; time for ≥1 retry
- LAN switches required.

Communication path characteristics
- Minimal delays
- Expected response time ≤ ReceiveTMO / 2

  (Otherwise ERROR when parameters are set)

4.5.5 Profile III (Medium & Cleanroom)

⚠ WARNING

Only the **noisy profiles** are suitable for safety-related process data communication.

Severe or fatal injuries.

Only use the noisy profiles for safety-related process data communication:
- Fast&Noisy, Medium&Noisy and Slow&Noisy

Application

The **Medium & Cleanroom** profile is for applications in an interference-free network that require only moderately fast data transmission.
- For medium data throughput
- Appropriate for virtual private networks (VPN) where data exchange is slow but error-free due to intermediate safety systems (firewalls, encryption).
- Appropriate for applications where the worst case reaction time is not a critical factor.

Network requirements
- Medium: 10 Mbit (10 BASE-T), 100 Mbit (100 BASE-TX), 1 Gbit technology
- LAN switches required.
- Clean: Interference-free network.

  Data losses due to network overload, outside influences and network manipulation must be avoided.

  Time for ≥ 0 retries

Communication path characteristics
- Moderate delays
- Expected response time ≤ ReceiveTMO

  (Otherwise ERROR when parameters are set)

4.5.6 Profile IV (Medium & Noisy)

Application

The **Medium & Noisy** profile is for applications that require only moderately fast data transmission.
- For medium data throughput
- For applications that require only moderately fast data transmission
- Appropriate for applications where the worst case reaction time is not a critical factor.

Network requirements
- Medium: 10 Mbit (10 BASE-T), 100 Mbit (100 BASE-TX), 1 Gbit technology
- LAN switches required.
- Noisy: Network is not interference-free.

  Low likelihood of data packet loss;

  time for ≥ 1 retry

Communication path characteristics
- Moderate delays
- Expected response time ≤ ReceiveTMO / 2

  (Otherwise ERROR when parameters are set)

4.5.7 Profile V (Slow & Cleanroom)

⚠ WARNING

Only the **noisy profiles** are suitable for safety-related process data communication.

Severe or fatal injuries.

Only use the noisy profiles for safety-related process data communication:
- Fast&Noisy, Medium&Noisy and Slow&Noisy

Application

The **Slow & Cleanroom** profile is for applications in an interference-free network that require only slow data transmission.
- For slow data throughput
- For applications that require only slow data transmission to (possibly far away) controllers and where the conditions of the communication path cannot be predicted.

Network requirements
- Slow: Data transfer via ISDN, dedicated line or radio relay connection.
- Clean: Interference-free network.

  Data losses due to network overload, outside influences and network manipulation must be avoided.

  Time for ≥ 0 retries

Communication path characteristics
- Moderate delays
- Expected response time ≤ ReceiveTMO

  (Otherwise ERROR when parameters are set)

4.5.8 Profile VI (Slow & Noisy)

Application

The **Slow & Noisy** profile is for applications that require only slow data transmission to (possibly far away) controllers.
- For slow data throughput
- For applications, mainly for data transfer via bad telephone wires or faulty radio relay paths.

Network requirements
- Slow: data transfer via telephone, satellite, radio, etc.
- Noisy: Network is not free of interference.

  Low likelihood of data packet loss;

  time for ≥ 1 retry

Communication path characteristics
- Moderate to long delays
- Expected response time ≤ ReceiveTMO / 2

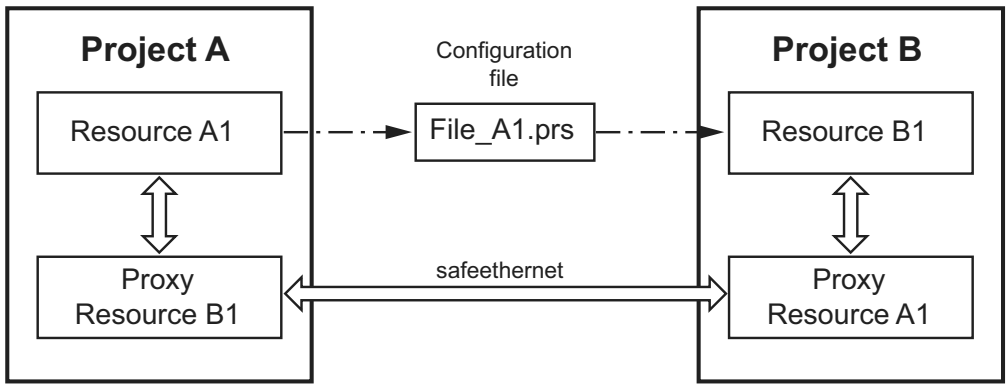
  (Otherwise ERROR when parameters are set)

4.6 Communication across projects

Communication across projects is used for the following:
- To connect resources from different projects with each other.
- To connect controllers with the SILworX operating system and controllers via safeethernet with each other.

The two projects communicate via safeethernet and the communication is configured in the safeethernet editor.

safeethernet connection between resource A1 in project A and resource B1 in project B:



5306777483

The project for which you configure the safeethernet connection and create the configuration file is called the local project.

The project into which you import the configuration file is called the target project.

During data exchange, the local and target projects are equal communication partners.

Each proxy resource serves as a placeholder for the corresponding resource from the external project and is used for importing and exporting the safeethernet connections.

The *Proxy Resource B1* in project A is the placeholder for the *Resource B1* in project B.

The *Proxy Resource A1* in project B is the placeholder for the *Resource A1* in project A.

You must create and configure the proxy resource (here *Proxy Resource B1*) manually in the local project (here *Project A*). After configuration, import the configuration file (here `File_A1.prs`) into the target project (here *Resource B1*).

The configuration file `File_A1.prs` contains the complete description of *Resource A1* for the safeethernet connection with *Resource B1*. After the configuration file `File_A1.prs` is imported into *Resource B1*, the *Proxy Resource A1* is automatically created in Project B.

4.6.1 Variants of communication across projects

In the two variants below, projects A and B communicate with each other via safeethernet.

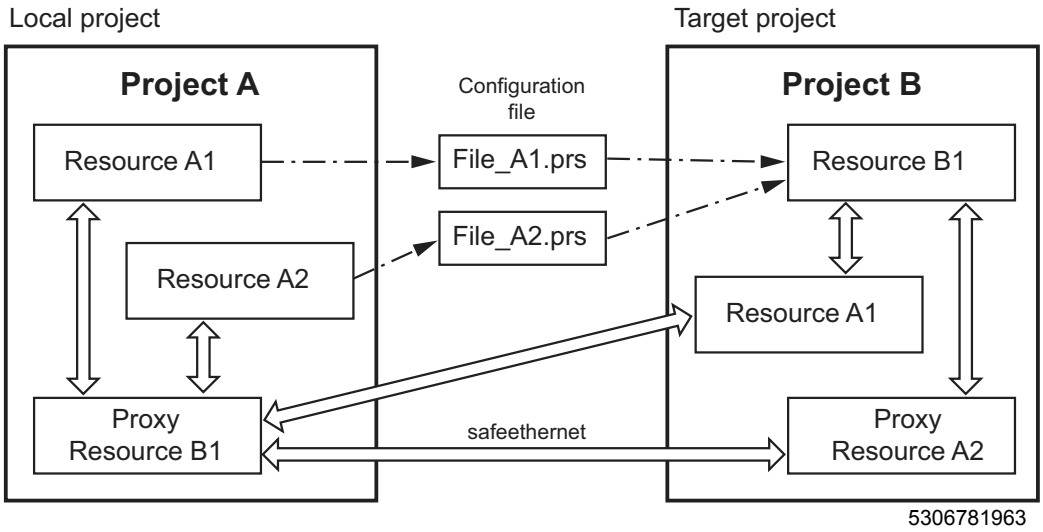
In the first variant Project A is the local project and in the second variant Project B is the local project. In general, the user may decide which of the two projects to create the configuration in.

The effort of configuration is about the same either way, and results in the same configuration.

Local project A

You can configure the communication with target project B and create the configuration files in local project A. The advantage is that only *Proxy Resource B1* must be created manually in the local project.

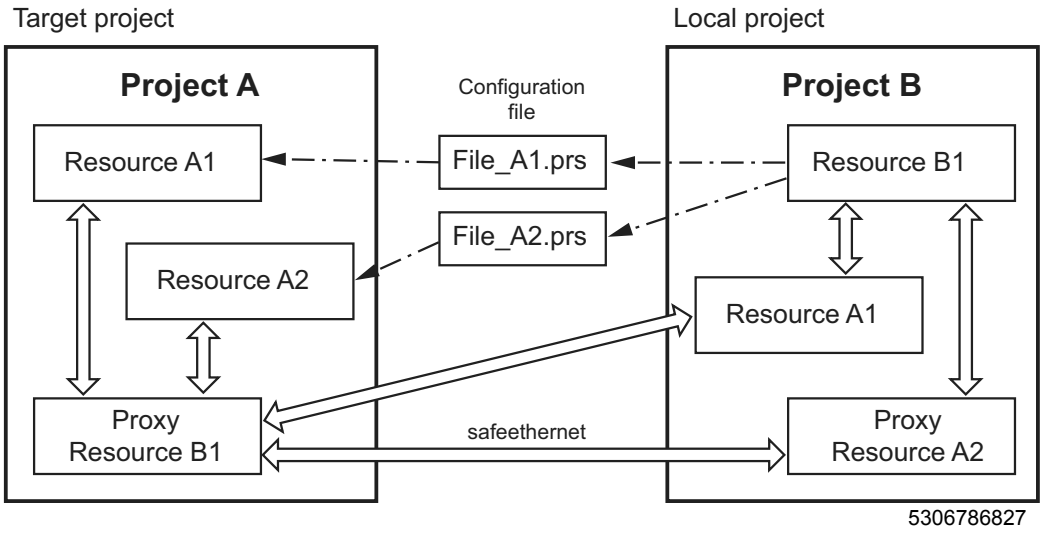
Variant with project A as local project:



Local project B

You can configure the communication with target project A and create the configuration files in local project B. The disadvantage is that you have to create two *Proxy Resources* (*A1* and *A2*) manually in local project B.

Variant with project B as local project:



4.7 Control panel (safeethernet)

The user can verify and control the settings for the safeethernet connection in the control panel. Current status information (e.g. cycle time, bus status, etc.) for the safeethernet connection is also displayed.

This is how you open the control panel to monitor the safeethernet connection:

1. Select [Resource] in the structure tree.
2. Select [Online] from the resource's context menu.
3. Enter access data in the system login to open the resource's control panel.

4. Select [safeethernet] in the control panel's structure tree.

5428388235

Resetting statistical values:

By using the context menu function you can reset the statistical data (cycle time min., max., etc.) to zero.

This is how you reset the statistical data of the safeethernet connection:

- Select safeethernet connection in the structure tree.
- Select [Reset safeethernet statistics] in the context menu of the safeethernet connection.

4.7.1 Display field (safeethernet connection)

The following values for the selected safeethernet connection are shown in the display field:

Element	Description
Name	Resource name of the communication partner
SRS	System.Rack.Slot
Connection status	Status of the safeethernet connection (see also the "Detail view of the safeethernet editor" section)
Receive timeout [ms]	See the "safeethernet parameters" section
Resend timeout [ms]	See the "safeethernet parameters" section
Acknowledge timeout [ms]	See the "safeethernet parameters" section
Min. RspT [ms]	Actual response time as minimum, maximum, last and average values (see the "safeethernet parameters" section).
Max. RspT [ms]	
Last RspT [ms]	
Average RspT [ms]	
Faulty messages	
Retries	Number of messages rejected since the statistics were reset.
Number of successful connections	Number of successful connections since the statistics were reset.
Early queue usage	Number of messages that were placed into the early queue since the statistics were reset (see the "safeethernet parameters" section).
Frame no.	Revolving sent counter
Ack. frame no.	Revolving receipt counter
Monotonicity	Revolving user data sent counter
Layout version	Signature of the current communications end point
New layout version	Signature of the new communications end point
Connection control	Status of the connection control
Channel 1 transmission control	Enabling Channel 1 transmission route (See the "safeethernet parameters" section)
Channel 2 transmission control	Enabling Channel 2 transmission route (See the "safeethernet parameters" section)
Quality of channel 1	Status of Channel 1 transmission route (See the "safeethernet parameters" section)
Quality of channel 2	Status of Channel 2 transmission route (See the "safeethernet parameters" section)
Redundant messages received late	For redundant transmission routes. Number of messages received late since the statistics were reset.

Element	Description
Lost redundant messages	For redundant transmission routes. Number of messages received on only one of the two transmission routes since the statistics were reset.
Protocol version	2: New protocol version for CPU operating systems of V7 or later

4.8 Maximum communications time slice

The maximum communications time slice is the allotted time in milliseconds (ms) per cycle during which the processor system processes the communications tasks. If not all communications tasks scheduled for a cycle can be processed, the complete communications data is transmitted over several cycles (number of communications time slices > 1).

NOTE

The condition that the number of communications time slices = 1 applies. The duration of the communications time slice must be set high enough that the cycle cannot exceed the watchdog time set by the process when using the entire communications time slice (see also the "Maximum response time for safeethernet" section).

4.9 Connections for safeethernet/Ethernet

The safety controller has the following interfaces for networking via safeethernet/ Ethernet:

The following interfaces are available:

- **2 Ethernet interfaces:** X4233_1 and X4233_2

  Both interfaces are located on the unit's terminal strip
- **1 Ethernet service interface:** X4223

  For connecting a programming unit (PADT)

The different systems can be freely linked to each other via Ethernet (star or linear configuration). A programming device (PADT) can also be connected at any location.

NOTE

Possible disruptions in Ethernet operation.

- Take care that no network rings are created during interconnection.
- Data packets can only reach a system on **one** path.

5 Modbus TCP/UDP

5.1 Modbus master

Data is transmitted between the Modbus master and the Modbus slaves via TCP/UDP (Ethernet).

The table below shows you the properties of the Modbus master:

Property	Description				
Modbus master	One Modbus master can be configured for each COM module/controller. The Modbus master can exchange data simultaneously • with TCP/UDP slaves				
Max. number of Modbus slaves	One Modbus master can serve up to 247 slaves. • 64 TCP slaves via TCP/IP connection • 247 UDP slaves via UDP/IP connection The maximum number of UDP slaves is limited because the slaves must be managed on the master side.				
Max. number of request messages	Up to 988 request messages can be configured for each Modbus master.				
Max. process data length per request message	The process data length for SEW-EURODRIVE-specific request messages is 1100 bytes; see the "SEW-EURODRIVE-specific function codes" section.				
Max. size of send data Max. size of receive data	64 kB send 64 kB receive Note: The status byte of the master and the status bytes of each associated slave must be subtracted from the max. size of the send data.				
Modbus data display format	The safety controller uses the big-endian format. Example: 32-bit data (e.g. DWORD, DINT):				
	32-bit data (hex)	0x12345678			
	Memory offset	0	1	2	3
	Big endian	12	34	56	78
	Middle endian	56	78	12	34
	Little endian	78	56	34	12

5.1.1 Adding a Modbus master

NOTE

If the Modbus master and the Modbus slave are located in different sub-networks, the corresponding user-defined routes must be entered into the routing table.

Configuring the Modbus TCP master

This is how you add the Modbus master:

1. Open [Configuration]/[Resource]/[Protocols] in the structure tree.
2. Select [New]/[Modbus master] in the context menu of Protocols to add a new Modbus master.
3. Select [Properties]/[General] in the context menu of the Modbus master.
4. Select [COM module].

   The remaining parameters retain the default values.

This is how you create the connection to a Modbus TCP slave in a Modbus master:

1. Open [Resource]/[Protocols]/[Modbus master]/[Ethernet slaves] in the structure tree.
2. Right-click on [Ethernet slaves] and select [New] in the context menu.
3. Choose "TCP/UDP slave" from the list and confirm with [OK].

4. Configuring the TCP/UDP slave in the Modbus master:
   - Select [Edit] to assign the system variables; see the "System variables gateway slave" section.
   - Select [Properties] to configure the properties; see the "Properties of gateway slave" section.

     Enter the IP address of the TCP/UDP slave in the properties of the slave.

     The remaining parameters retain the default values.

5.1.2 Modbus master menu functions

Edit

The Modbus master's "Edit" dialog window contains the following tab:

System variables

The "System variables" tab provides system variables that enable users to analyze the status of the Modbus master in the user program and control the Modbus master.

Element	Description
Number of faulty slave connections	Number of faulty connections with Modbus slaves whose status is "activated". Deactivated Modbus slaves are not taken into account here.
Modbus master activation control	This allows the Modbus master to be stopped or started by the user program. 0: Activate 1: Deactivate (edge-triggered – Modbus master can be activated via PADT even if Modbus master activation control = 1.)
Modbus master bus error	Bus error, e.g. message error (unknown codes, etc.), length error.
Modbus master status	The Modbus master status displays the current protocol status: 1: OPERATE 0: OFFLINE
Resetting all slave errors	By switching from FALSE->TRUE all slave errors and bus errors are reset.

Properties

The "Properties" menu function in the Modbus master's context menu opens the Properties dialog.

The dialog contains the following registers:

General

The name and description for the Modbus master are entered in the "General" register. In addition, parameters are set here if the Modbus master will also function as a TCP and/or UDP gateway.

Parameter	Description
Type	Modbus master
Name	Name of the Modbus master
Module	Selecting the COM module where this protocol is processed.
Activate max. µP budget	**Activated:** Copy µP budget limit from the "Max. µP budget in [%]" field. **Deactivated:** Do not use a µP budget limit for this protocol.
Max. µP budget in [%]	Maximum µP load for the module which can be produced while processing the protocol. Value range: 1 to 100% Default value: 30%
Behavior if CPU/COM connection is lost	If the connection between the processor module and the communications module is lost, the input variables are either initialized or used unchanged in the processor module, based on this parameter. (For example, if the communications module is disconnected while communication is in progress.) **Accept initial data:** Input variables are reset to the initial values. **Maintain last value:** Input variables retain the last value.
Activate TCP gateway	This function cannot be activated because the RS485 interface is being used by the com-user task.
TCP server port	Default: 502 Other TCP ports can also be configured. In this case, the port assignment at the *Internet Corporation for Assigned Names and Numbers (ICANN)* must be observed.
Maximum number of TCP connections as servers	Maximum number of TCP connections as servers which can be opened simultaneously Value range: 1 to 64 Default value: 5

Parameter	Description
Activate UDP gateway	This function cannot be activated because the RS485 interface is being used by the com-user task.
UDP port	Default: 502 Other TCP ports can also be configured. In this case, the port assignment at the *Internet Corporation for Assigned Names and Numbers (ICANN)* must be observed.
Maximum queue length	Length of the gateway queue for unanswered request messages from other masters. This is only observed if a gateway has been activated. Value range: 1 to 20 Default value: 3

CPU/COM

The default values for the parameters provide the fastest possible exchange of Modbus data between the COM module and the CPU module in the safety controller.

These parameters should only be changed if a reduction of the COM and/or CPU load is required for an application and the process permits this.

NOTE

Changing the parameters is only recommended for experienced programmers. Increasing the COM and CPU update time also means that the actual update time of the Modbus data is increased.

- The time requirements of the system must be checked.

Parameter	Description
Process data update interval [ms]	Update time in milliseconds during which the protocol data is exchanged between COM and CPU. If the process data update interval is zero or less than the controller's cycle time, the data exchange occurs as quickly as possible. Value range: 0 to (2^{31}-1) Default value: 0
Force process data consistency	**Activated:** Transfers all protocol data from the CPU to the COM within a single CPU cycle. **Deactivated:** Transfers all protocol data from the CPU to the COM, distributed across several CPU cycles at 1100 bytes per data direction. This may also reduce the cycle time of the controller. Default value: Activated

5.1.3 Modbus function codes of the master

With the Modbus function codes (request messages), you can write or read variables in both directions. Individual variables or several consecutive variables can be read or written.

This is how you create a new request message for a TCP/UDP slave:

1. Select a TCP/UDP slave in [Resource]/[Protocols]/[Modbus master]/[Ethernet slaves] in the structure tree.
2. Right-click on TCP/UDP slave and select [New] in the context menu.
3. Select a request message from the "New object" dialog.

Modbus default function codes

The Modbus master supports the following Modbus standard function codes.

Element	Code	Type	Meaning
READ COILS	01	BOOL	Reading multiple variables (BOOL) from the slave.
READ DISCRETE INPUTS	02	BOOL	Reading multiple variables (BOOL) from the slave.
READ HOLDING REGISTERS	03	WORD	Reading multiple variables of any type from the slave.
READ INPUT REGISTERS	04	WORD	Reading multiple variables of any type from the slave.
WRITE SINGLE COIL	05	BOOL	Writing a single signal (BOOL) to the slave.
WRITE SINGLE REGISTER	06	WORD	Writing a single signal (WORD) to the slave.
WRITE MULTIPLE COILS	15	BOOL	Writing multiple variables (BOOL) to the slave.
WRITE MULTIPLE REGISTERS	16	WORD	Writing multiple variables of any type to the slave.
READ WRITE HOLDING REGISTERS	23	WORD	Writing and reading multiple variables of any type to and from the slave.

NOTE

Additional information about Modbus is available in the "Modbus Application Protocol Specification" at *www.modbus.org*

SEW-EURODRIVE-specific function codes

The SEW-EURODRIVE-specific function codes correspond to the default Modbus function codes. The two differences are the maximum allowable process data length of 1100 bytes and the format of request and response headers.

Element	Code	Type	Meaning
Read Coils Extended	100 (0x64)	BOOL	Corresponds to function code 01 Reading multiple variables (BOOL) from the slave's import or export section. Maximum length of the process data: 1100 bytes
Read Discrete Inputs Extended	101 (0x65)	BOOL	Corresponds to function code 02 Reading multiple variables (BOOL) from the slave's import or export section. Maximum length of the process data: 1100 bytes
Read Holding Registers Extended	102 (0x66)	WORD	Corresponds to function code 03 Reading multiple variables (BOOL) from the slave's import or export section. Maximum length of the process data: 1100 bytes
Read Input Registers Extended	103 (0x67)	WORD	Corresponds to function code 04 Reading multiple variables (BOOL) from the slave's import or export section. Maximum length of the process data: 1100 bytes
Write Multiple Coils Extended	104 (0x68)	BOOL	Corresponds to function code 15 Writing multiple variables (BOOL) to the slave's import section. Maximum length of the process data: 1100 bytes
Write Multiple Registers Extended	105 (0x69)	WORD	Corresponds to function code 16 Writing multiple variables (BOOL) to the slave's import section. Maximum length of the process data: 1100 bytes

Element	Code	Type	Meaning
Read/Write Multiple Registers Extended	106 (0x6A)	WORD	Corresponds to function code 23 Writing and reading multiple variables of any type to and from the slave's import or export section. Maximum length of the process data: 1100 bytes (request message from the Modbus master) 1100 bytes (response to the master).

5.1.4 Format of the request and response headers

The request and response headers of the SEW-EURODRIVE-specific Modbus function codes are structured as follows:

Code	Request	Response
100 (0x64)	1 byte function code 0x64 2 bytes start address 2 bytes number of coils 1 – 8800(0x2260)	1 byte function code 0x64 2 bytes number of bytes = N N bytes of coil data (8 coils are packed into one byte)
101 (0x65)	1 byte function code 0x65 2 bytes start address 2 bytes number of discrete inputs 1 – 8800(0x2260)	1 byte function code 0x65 2 bytes number of bytes = N N bytes discrete inputs – data (8 discrete inputs are packed into one byte)
102 (0x66)	1 byte function code 0x66 2 bytes start address 2 bytes number of registers 1 – 550(0x2260)	1 byte function code 0x66 2 bytes number of bytes = N N bytes register data
103 (0x67)	1 byte function code 0x67 2 bytes start address 2 bytes number of registers 1 – 550(0x2260)	1 byte function code 0x67 2 bytes number of bytes = N N bytes register data
104 (0x68)	1 byte function code 0x68 2 bytes start address 2 bytes number of coils 1 – 8800(0x2260) 2 bytes number of bytes = N N bytes coil data	1 byte function code 0x68 2 bytes start address 2 bytes number of coils 1 – 8800(0x2260)
105 (0x69)	1 byte function code 0x69 2 bytes start address 2 bytes number of registers 1 – 550(0x226) 2 bytes number of bytes = N N bytes register data	1 byte function code 0x69 2 bytes start address 2 bytes number of registers 1 – 550(0x226)
106 (0x6A)	1 byte function code 0x6a 2 bytes reading start address 2 bytes number of read registers 1 – 550(0x226) 2 bytes writing start address 2 bytes number of write registers 1 – 550(0x226) 2 bytes number of bytes for writing = N N bytes register data	1 byte function code 0x6a 2 bytes number of bytes = N N bytes register data

5.1.5 Read request messages

Variables can be read from the slave with the read function codes. A Modbus master request message includes the start address of the read/write section and the Modbus function.

The Modbus master sends a read request message to the Modbus slave in order to read the variables. The Modbus slave then returns a response message with the requested variables to the Modbus master.

This is how you configure a read request message:

1. In the structure tree, select [Request message] for configuration.
2. Right-click on "Request message" and select [Edit] in the context menu.
3. In the object selection, select a global variable to be used as Modbus receiving variable and drag & drop the variable to an empty spot in the input signal area.
4. Repeat this step for each additional Modbus receiving variable.
5. Open the context menu by right-clicking on an empty spot in the "Input signals" area and select [New offsets] to renumber the offsets of the variables.

The following read request messages are available:

Read Coils (01) and Extended (100)

Reading multiple variables (BOOL) from the slave.

Element	Meaning
Type	Modbus function: Read Coils
Name	Any unique name for the Modbus function
Description	Description of the Modbus function
Start address of the read section	0 – 65535

Read Discrete Inputs (02) and Extended (101)

Reading multiple variables (BOOL) from the slave.

Element	Meaning
Type	Modbus function: Read Discrete Inputs
Name	Any unique name for the Modbus function
Description	Description of the Modbus function
Start address of the read section	0 – 65535

Read Holding Registers (03) and Extended (102)

Reading multiple variables of any type from the slave.

Element	Meaning
Type	Modbus function: Read Holding Registers
Name	Any unique name for the Modbus function
Description	Description of the Modbus function
Start address of the read section	0 – 65535

Read Input Registers (04) and Extended (103)

Reading multiple variables of any type from the slave

Element	Meaning
Type	Modbus function: Read Input Registers
Name	Any unique name for the Modbus function
Description	Description of the Modbus function
Start address of the read section	0 – 65535

5.1.6 Read and write request messages

The Modbus master sends a read and write request message to the Modbus slave in order to read and write the variables.

The Modbus master first writes the specified write variables to the specified import section on the Modbus slave.

The Modbus master then reads the specified read variables from the specified export section on the Modbus slave.

NOTE

The write and read functions are independent of each other even in read and write request messages; they are simply sent in the same request message.

However, read and write request messages are often used to read the written variables back to the Modbus master. This checks whether the variables sent were written correctly.

This is how you configure a read and write request message:

1. In the structure tree, select [Request message] for configuration.
2. Right-click on "Request message" and select [Edit] in the context menu.

This is how you configure the read variables:

1. In the object selection, select a global variable to be connected with the new Modbus receiving variable and drag and drop that global variable into the "Global variable" column of the Modbus receiving variables.
2. Repeat step 1 for each additional Modbus receiving variable.
3. Open the context menu by right-clicking on an empty spot in the "Input signals" area and select [New offsets] to renumber the offsets of the variables.

This is how you configure the write variables:

1. In the object selection, select a global variable to be connected with the new Modbus sending variable and drag and drop that global variable into the "Global variable" column of the Modbus sending variables.
2. Repeat step 1 for each additional Modbus sending variable.
3. Open the context menu by right-clicking on an empty spot in the "Output signals" area and select [New offsets] to renumber the offsets of the variables.

Read Write Holding Register (23) and Extended (106)

Writing and reading multiple variables of any type to and from the slave's import section.

Element	Meaning
Type	Modbus function: *Read Write Holding Registers*
Name	Any unique name for the Modbus function
Description	Description of the Modbus function
Start address of the read section	0 – 65535
Start address of the write section	0 – 65535

5.1.7 Write request message

With the write function codes, variables are only written to a slave's import area.

A Modbus master request message includes the start address of the read/write section and the Modbus function.

The Modbus master sends a write request message to the Modbus slave in order to write variables. The Modbus slave writes the variables received to its import section.

The variables which the Modbus master writes to the Modbus slave must be inserted in a write request message's "Assign variables" dialog.

This is how you configure a write request message:

1. In the structure tree, select [Request message] for configuration.
2. Right-click on "Request message " and select [Edit] in the context menu.
3. In the object selection, select a global variable to be used as Modbus sending variable and drag and drop that variable to an empty spot in the "Send signals" area.
4. Repeat step 3 for each additional Modbus sending variable.
5. Open the context menu by right-clicking on an empty spot in the "Send signals" area and select [New offsets] to renumber the offsets of the variables.

The following write request messages are available:

Write Multiple Coils (15) and Extended (104)

Writing multiple variables (BOOL) to the slave's import section.

Element	Meaning
Type	Modbus function: *Write Multiple Coils*
Name	Any unique name for the Modbus function
Description	Description of the Modbus function
Start address of the write section	0 – 65535

Write Multiple Registers (16) and Extended (105)

Writing multiple variables of any type to the slave's import section.

Element	Meaning
Type	Modbus function: *Write Multiple Registers*
Name	Any unique name for the Modbus function
Description	Description of the Modbus function
Start address of the write section	0 – 65535

Write Single Coil (05)

Writing a single variable (BOOL) to the slave's import area.

Element	Meaning
Type	Modbus function: *Write Single Coil*
Name	Any unique name for the Modbus function
Description	Description of the Modbus function
Start address of the write section	0 – 65535

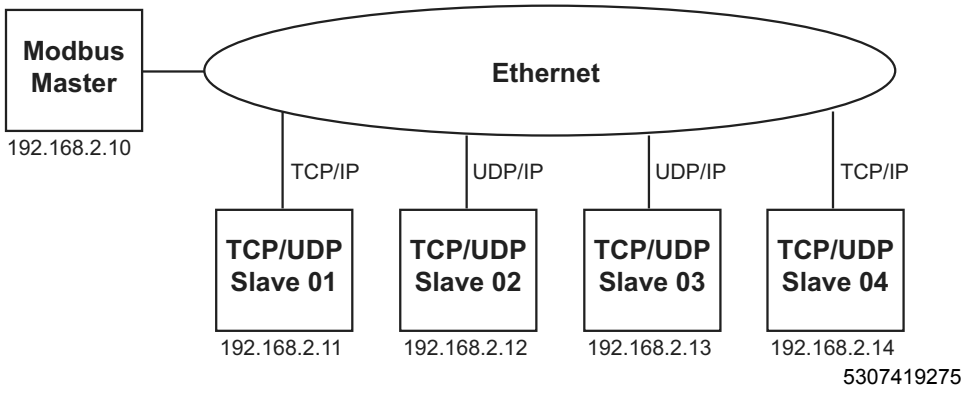
Write Single Register (06)

Writing a single variable (WORD) to the slave's import area.

Element	Meaning
Type	Modbus function: *Write Single Register*
Name	Any unique name for the Modbus function
Description	Description of the Modbus function
Start address of the write section	0 – 65535

5.1.8 Ethernet slaves (TCP/UDP slaves)

The Modbus master can communicate with up to 64 TCP/IP and 247 UDP/IP slaves.



This is how you create a new connection to a TCP/UDP slave in the Modbus master:

1. Open [Resource]/[Protocols]/[Modbus master]/[Ethernet slaves] in the structure tree.
2. Right-click on [Ethernet slaves] and select [New] in the context menu.
3. Select "TCP/UDP slaves" from the list and confirm with [OK].
4. Configuring the TCP/UDP slave in the Modbus master:

   Select [Edit] to assign the system variables; see the "System variables of TCP/UDP slaves" section.

   Select [Properties] to configure the properties; see the "Properties of TCP/UDP slaves" section.

NOTE

If the TCP/UDP slaves and the Modbus master are located in different sub-networks, the corresponding user-defined routes must be entered into the routing table.

In addition to the IP address, the Modbus TCP master always sends a Modbus slave address (unit identifier) in its messages to the Modbus TCP slave. This address is always FF_{Hex} (255).

System variables of the TCP/UDP slaves

The system variables register provides system variables that allow users to analyze and control the status of the TCP/UDP slave in the user program.

The TCP/UDP slave status can be analyzed in the user program using the following status variables:

Element	Description	
Modbus slave activation control	This allows the user program to deactivate or activate the TCP/UDP slave.	0: Activate 1: Deactivate (edge-triggered – Modbus slave can be activated via PADT even if Modbus slave activation control = 1.)
Modbus slave error	Error code	The error codes 0x01 – 0x0b correspond to the exception codes of the Modbus protocol specification. 0x00: No error
	Exception codes:	0x01: Invalid function code 0x02: Invalid addressing 0x03: Invalid data 0x04: (Not used) 0x05: (Not used) 0x06: Device busy (gateway only, not supported) 0x08: (Not used) 0x0a: (Not used) 0x0b: No response from slave (gateway only, not supported)
	SEW-EURODRIVE-specific codes	0x10: Faulty frame received 0x11: Frame with wrong transaction ID received 0x12: Unexpected response received 0x13: Response received via wrong connection 0x14: Wrong response to a write order 0xff: Slave timeout
Modbus slave status	Connection status of the TCP/UDP slave	0: Deactivated 1: Not connected 2: Connected

Properties of the TCP/UDP slaves

The following parameters must be set in the Modbus master to configure the connection to the TCP/UDP slave:

Parameter	Description
Type	TCP/UDP slave
Name	Any unique name for the TCP/UDP slave
Description	Any unique description for the TCP/UDP slave
Master-slave data exchange [ms]	Interval for data exchange with this slave, 1 through (2^{31}-1). If the slave could not be reached after the maximum number of send retries, the master-slave data exchange interval is increased by a factor of four.
TCP connection only if necessary	If the transport protocol is TCP, this setting determines whether the connection to this slave should be automatically terminated after each data exchange. TRUE: Terminate the connection. FALSE: Do not terminate the connection. Default value: FALSE
Receive timeout [ms]	Receive timeout for this slave [ms]. After this time has elapsed, a new send attempt is started.
IP address	IP address of the TCP/UDP slave
Port	Default: 502 Other TCP/UDP ports can also be configured. In this case, the port assignment at the *Internet Corporation for Assigned Names and Numbers (ICANN)* must be observed.
IP protocol communication type	TCP or UDP Default value: TCP

Parameter	Description
Maximum number of send retries	Maximum number of send retries if the slave doesn't answer. The number of send retries can be set freely (0 to 65535). Always zero for TCP/IP; cannot be changed. We recommend setting zero to eight send retries.

5.1.9 Control panel (Modbus master)

The user can verify and control the settings for the Modbus master in the control panel. Current status information (e.g. master status, etc.) for the master is also displayed.

This is how you open the control panel to monitor the Modbus master:

1. Select [Hardware] in the structure tree and [Online] in the context menu.
2. Enter access data into the system login to open the online view of the hardware.
3. Double-click on "COM module" and select [Modbus master] in the structure tree.

Context menu (Modbus master)

The following commands can be selected from the context menu of the selected Modbus master:

Offline: This command stops the Modbus master.

Operate: This command starts the Modbus master.

Reset statistics: Resets the statistical data (e.g. number of bus errors, cycle time min., max., etc.) to zero.

Display field (Modbus master)

The following values of the selected Modbus master are shown in the display field:

Element	Description
Name	Name of the Modbus master
Master status	The Modbus master status displays the current protocol status: OPERATE OFFLINE
Number of bus errors	Counter for number of bus errors
Interrupted connections	Counter for number of interrupted connections
µP load (planned)	See properties in the "Modbus master menu functions" section
µP load (actual)	

5.1.10 Control panel (Modbus master->slave)

The user can verify and activate/deactivate the settings for the Modbus master's communication partners in the control panel.

Current status information (e.g. slave status, etc.) for the communication partner is also displayed.

This is how you open the control panel to monitor the Modbus connection:

- Select [Hardware] in the structure tree and [Online] in the context menu.
- Enter access data into the system login to open the online view of the hardware.
- Double-click on "COM module" and select [Modbus master]/[Slave] in the structure tree.

6 Com-user task (CUT)

In addition to the user program created with SILworX, a C program can also be run on the controller. This non-secure C program runs as a com-user task and does not impact the secure processor module of the controller's communications module.

The com-user task has its own cycle which is independent of the CPU cycle.

6.1 CUT features

The following table describes the features of the CUT

Element	Description
Com-user task	One com-user task can be configured for each safety controller.
Safety-related	No

6.2 Requirements

You will need the following in order to set up a SILworX program with a com-user task:

- Firmware:

  CUT PFF-HM31, part number: 28202430.xx

  Refer to the "Com-User Task for PFF-HM31A" manual for information on this topic.

- Software that is **not** included in the scope of delivery:

  You can obtain this software and documentation from SEW-EURODRIVE on CD or DVD by providing the following order information:

Designation	Item number
SILWorX for PFF-HM31A • Hardware: SILWorX license dongle • Software: SILWorX 4.64.0 or later	1 950 011 4
PFF-HM31 Motion Library Function block library for safety-related position detection	1 710 640 0

- You will need the MOVIVISION® Configuration and Diagnostics Tool Version 2.0 software for diagnostics for com-user task applications (not included in the scope of delivery).

7 Operating system

The operating system includes all basic safety controller functions.

The user functions that each PES is to execute are set in the user program. A code generator translates the user program into machine code. The programming tool transfers this machine code into the controller's flash memory.

7.1 Processor operating system functions

The main functions of the processor system's operating system and the connections to the user program are shown in the table below.

Operating system functions	Connections to the user program
Cyclical processing of the user program.	Acts on variables, function modules.
Configuration of the programmable controller.	Set by selecting the controller.
Processor tests.	-
I/O module tests.	-
Responses in case of error.	Fixed. The user program is responsible for process responses.
Diagnostics for processor system and inputs/ outputs.	Use of system signals/variables for error messages.
Secure communication: • Peer-to-peer Non-secure communication: • Modbus	Defining the use of communications signals/variables.
PADT interface: • Permitted actions	Set in the programming tool: • Configuration of protective functions • User login

Each operating system is verified by the relevant TÜV (German Technical Control Board) and approved for operation with safety-related controllers. Each valid version of the operating system and its associated signatures (CRCs) are documented in a list that SEW-EURODRIVE creates together with the German Technical Control Board.

7.2 Behavior if errors occur

The response to errors which have been identified by tests is important. The following types of errors can occur:

- Permanent input and output errors
- Temporary input and output errors
- Internal errors

7.2.1 Permanent input and output errors

An error that occurs in an input or output channel has no effect on the controller. The operating system only regards the channel as faulty, not the entire controller. The other safety functions are not affected by this and remain active.

In case of faulty input channels, the operating system transfers the secure value "0" or the initial value to processing.

The operating system switches off faulty output channels. If it is not possible to turn off only one channel, the entire output module is regarded as faulty.

The operating system sets the error status signal and notifies the user program of the error type.

If the controller cannot turn off the corresponding output and the 2nd path for turning off the output is also ineffective, the controller switches to STOP. The processor system's watchdog then turns off the outputs.

If errors in the I/O modules have been open for more than 24 hours, the controller permanently turns off only the corresponding I/O modules.

7.2.2 Temporary input and output errors

If an error occurs in an input or output module and disappears on its own, the operating system resets the error status and resumes regular operation.

The operating system statistically analyzes the frequency with which errors occur. It sets the module status to permanently faulty if the configured error frequency is exceeded. This causes the module to stop working even after the error disappears. When the controller's operating status is changed from STOP to RUN, the module is enabled and the error statistic is deleted. This change acknowledges the error in the module.

7.2.3 Internal errors

NOTE

On the rare occasion that a safety controller discovers an internal error, the following error response will occur:

The safety controller is automatically rebooted.

If an internal error occurs again within one minute of booting, the safety controller remains in the STOP/INVALID CONFIGURATION status.

8 User program

The PES user program must be created and loaded using a programming device on which the SILworX programming tool has been installed according to the requirements of IEC 61131-3.

The user program must first be created and configured for safety-related operation of the controller using the programming tool. The requirements in the "PFF-HM31A Decentralized Safety Controller for MOVIPRO®" safety manual must be met.

After the program has been compiled, the programming device loads the user program (logic) and the configuration (connection parameters such as IP address, subnet mask and system ID) into the controller and starts the controller.

The programming device offers the following options for working with the controller while the controller is in operation:

- Starting and stopping the user program
- Displaying and forcing variables/signals with the force editor
- Step-by-step execution of the user program in test mode – not in safety-related operation
- Reading the diagnostics history

The programming device must have the same user program as the controller for the diagnostics history to be read.

The following optional functions are available for the user program:

- **Multitasking:**

  The ability of the safety controller to process up to 32 user programs within the processor module is called multitasking.

  This allows for separation of sub-functions within a single project. Individual user programs can be started, stopped and reloaded independently of each other.

- **Reload:**

  If changes were made to the user programs, these changes can be transferred to the PES while it is in operation. The operating system verifies and activates the modified user program, which then assumes the control function.

NOTE

In unit option PFF-HM31A1-E61-I111-00/000/000, the optional functions can be used for testing purposes without activation for 5,000 operating hours. The "ERROR" system LED lights up red continuously when functions that were not enabled are used.

After the 5,000 operating hours have elapsed, the controller no longer starts.

- Order the unit option with the required functions in a timely manner.

8.1 User program operating modes

Only one user program at a time can be loaded onto each controller. The user program can be in one of the following operating modes:

Operating mode	Description
RUN	The processor system is in the RUN operating mode. The user program is executed cyclically; I/O signals are processed.
Test mode (individual step)	The processor system is in the RUN operating mode. The user program is executed cyclically by manual request; I/O signals are processed. **Not allowed for safety-related operation.**
STOP	The processor system is in the STOP operating mode. The user program is not (or no longer) executed; the outputs are reset.
ERROR	A loaded user program has been stopped because of an error. The outputs are reset. Note: The program can only be restarted by the PADT.

8.2 General information about forcing

Forcing means replacing a variable's current value with a force value. A variable can receive its current value from the following sources:

- From a physical input
- From the communication
- From a logical link

When a variable is forced, the user sets the value. Forcing is used under the following circumstances:

- Testing the user program, particularly for rare cases that cannot be tested otherwise.
- Simulation of unavailable sensors in cases where the initial value is not appropriate.

⚠ WARNING

Personal injury may result from forced values.

Severe or fatal injuries may result.

- Only force values after consulting the inspection authority responsible for approving the system.
- Remove limitations to forcing only after consultation with the inspection authority responsible for approving the system.

During forcing, the person in charge must ensure sufficiently safe monitoring of the process through other technical and organizational measures. SEW-EURODRIVE recommends forcing only for a limited time.

⚠ WARNING

Disruption of safety-related operation may result from forced values.

Severe or fatal injuries may result.

- Forced values can cause incorrect output values.
- Forcing increases the cycle time. This can cause the watchdog time to be exceeded.

Forcing is only permitted after consultation with the inspection authority responsible for approving the system.

Basic information about forcing is provided in the "Maintenance Override" document from the German Technical Control Board. The document is available on the following German Technical Control Board sites:

http://www.tuv-fs.com or http://www.tuvasi.com.

8.3 Forcing

Forcing can occur on two levels:

- Global forcing

  Global variables are forced for all applications
- Local forcing

  The values of local variables in a user program are forced

The following conditions must be met for a global or local variable to be forced:

- The corresponding force switch has been set
- Forcing has been started

If forcing has been started, a change to the force switch takes effect immediately. If forcing has been started and the force switch has been set, a change to the force value takes effect immediately. Local forcing can be started and stopped separately for each user program.

8.3.1 Time limit

Different time limits can be set for global and local forcing. After the time set has elapsed, the controller ends the forcing. The behavior of the safety controller after the time limit has elapsed can be configured.

- For global forcing, the following settings can be selected:
  - The resource stops
  - The resource continues to run
- For local forcing, the following settings can be selected:
  - The user program stops
  - The user program continues to run

It is also possible to force without a time limit. In this case, forcing must be ended manually. After forcing of a variable ends, the process value applies again.

8.3.2 Force editor

The SILworX force editor displays all variables which can be forced. Global and local variables are displayed separately in different registers. Force values and force switches can be set in the registers.

8.3.3 Restrictions on forcing

To avoid possible disruptions to safety-related operation by improper forces, the following measures can be taken in the configuration to restrict the use of forcing:

- Setting up different user accounts with and without forcing rights
- Prohibiting global forcing for a resource
- Prohibiting local forcing or entering of process values
- In addition, forcing can be turned off immediately with a keyswitch. For such a switch to take effect, the *Deactivate forcing* system variable must be connected to a digital input to which a keyswitch is connected.

  The *Deactivate forcing* system variable prevents the start of forcing for global and local variables and immediately turns off forces which have already been started.

9 Startup

Starting safety controller operations includes the following stages:

- Mechanical installation. Please observe the "Mechanical installation" section in the "PFF-HM31A Decentralized Safety Controller" operating instructions
- Electrical installation. Please observe the "Electrical installation" section in the "PFF-HM31A Decentralized Safety Controller" operating instructions
- Configuration
  - Creating the user program
  - Setting safety, communication and other parameters

9.1 Checklist for project planning, programming and startup

This checklist is a recommendation to the user

- About project planning, programming and startup of safety-related inputs and outputs
- About creating a user program with the SILworX programming tool

Filling in the checklist can ensure that requirements have been fully accounted for in an orderly format. The checklist is also a documentation of the connection between external wiring and the user program.

The *PFF_HM31_Checkliste.pdf* checklist can be downloaded as a PDF document from the SEW-EURODRIVE homepage (www.sew-eurodrive.com) in the "safetyDrive" area of the "Documentations" category.

9.2 Configuration with SILworX

The SILworX programming tool hardware editor displays the PFF-HM31A similarly to a base carrier, equipped with the following modules:

- Processor module (CPU)
- Communications module (COM)
- Digital input module (DI 26)
- Digital output module (DO 8)
- Counter module (HSC 2)

Double-clicking on the module opens the detail view with registers. In the registers, the global variables configured in the user program can be assigned to the system variables of the relevant module.

9.2.1 Processor module

Double-clicking on the modules opens the detail view with registers. In the registers, the global variables configured in the user program can be assigned to the system variables of the relevant module.

Module register

The module register includes the following parameters:

Parameter	Description
Name	Module name
Use max. µP budget for HH protocol	• Activated: Copy CPU load limit from the *Max. µP budget for HH protocol [%]* field. • Deactivated: Do not use a CPU load limit for safeethernet. Default setting: deactivated

Parameter	Description
Max. µP budget for HH protocol [%]	Maximum CPU load for the module that can be produced while processing the safeethernet protocol. **Note:** The maximum load must be distributed among all protocols which use this communications module.
IP address	IP address of the Ethernet interface. Default value: 192.168.0.99
Subnet mask	32-bit address mask for subdividing an IP address into network and host addresses. Default value: 255.255.252.0
Default interface	Activated: Interface is used as the default interface for a system login. Default setting: Deactivated
Default gateway	IP address of the default gateway. Default value: 0.0.0.0
ARP aging time [s]	A CPU or COM module saves the MAC addresses of its communication partners in a MAC/IP address assignment table (ARP cache). If during a period of 1× to 2× *ARP aging time* • Messages arrive from the communication partner, the MAC address remains in the ARP cache. • No messages arrive from the communication partner, the MAC address is deleted from the ARP cache. A typical value for the *ARP aging time* in a local network is 5 s to 300 s. The content of the ARP cache cannot be read by the user. When routers or gateways are used, the *ARP aging time* should be adjusted (increased) for additional delays in transmissions back and forth. If the *ARP Aging Time* is too low, the CPU/COM module deletes the MAC address of the communication partner from the ARP cache and communication is either delayed or dropped. For efficient use, the *ARP aging time* must be greater than the *ReceiveTimeouts* of the protocols used. Value range: 1 s to 3600 s Default value: 60 s
MAC learning	ARP cache learning behavior: • Conservative: MAC addresses of stored ARP entries are not overwritten by received messages. • Liberal: MAC addresses of stored ARP entries are overwritten by received messages. Default setting: conservative
IP forwarding	Allows a processor module to work as a router and forward data packets to other network nodes. Default setting: deactivated
ICMP mode	Internet Control Message Protocol (ICMP) message types supported by the processor module: • No ICMP responses • Echo response • Host unavailable • All implemented ICMP responses Default setting: echo response
Max. com. time slice ASYNC [ms]	Maximum value in ms of the time slice that is used for communication during the resource's cycle. Setting range: 2 to 5000 ms
Max. duration of configuration connections [ms]	Defines how much time is available during a CPU cycle for process data communication. Setting range: 6 to 5000 ms
Target cycle time [ms]	Desired or maximum cycle time, see *Target cycle time mode*. The *Target cycle time* cannot exceed the configured watchdog time (6 ms); otherwise the PES rejects it. Setting range: 0 to 7500 ms

Parameter	Description
Target cycle time mode	Using the *Target cycle time [ms]*. **Fixed:** The PES adheres to the *Target cycle time* and, if necessary, extends the cycle. This does not apply if the user programs' processing time exceeds the *Target cycle time*. **Fixed/liberal:** Same as fixed, but during the first activation cycle of the reload function (function available as a unit option), the *Target cycle time* is not observed. **Dynamic/liberal:** The PES adheres to the *Target cycle time* if possible, but executes the cycle in the shortest possible time. During the first activation cycle of the reload function (function available as a unit option), the *Target cycle time* is not observed.
Maximum system bus latency time [µs]	May not be used for the PFF-HM31A safety controller.
safeethernet CRC	**Current version:** The CRC for safeethernet is created with the current algorithm.

Routings register

The routings register contains the following parameters:

Parameter	Description
Name	Name of the routing setting
IP address	Target IP address of the communication partner (for direct host routing) or network address (for subnet routing). Value range: 0.0.0.0 to 255.255.255.255 Default value: 0.0.0.0
Subnet mask	Specified target address range for a routing entry. 255.255.255.255 (for direct host routing) or subnet mask of the subnet being addressed. Value range: 0.0.0.0 to 255.255.255.255 Default value: 255.255.255.255
Gateway	IP address of the gateway to the network being addressed. Value range: 0.0.0.0 to 255.255.255.255 Default value: 0.0.0.1

Ethernet switch register

The Ethernet switch register contains the following parameters:

Parameter	Description
Name	Name of the port (Eth1 through Eth4) as printed on the housing; only one configuration can exist for each port.
Speed [Mbit/s]	10 Mbit/s: data rate of 10 Mbit/s 100 Mbit/s: data rate of 100 Mbit/s 1000 Mbit/s: data rate of 1000 Mbit/s (not supported) Autoneg: baud rate is automatically set Default value: autoneg
Flow control	Full duplex: communication in both directions at once Half duplex: communication in one direction Autoneg: automatic communications control Default value: autoneg
Autoneg also for fixed values	*Advertising* (transmission of speed and flow control properties) is also performed if the values for *Speed* and *Flow control* are fixed. This allows other units whose ports are set to *Autoneg* to detect the setting of the safety controller's ports.
Limit	Limit incoming multicast and/or broadcast packets. Off: no limit Broadcast: limit broadcast (128 kbit/s) Multicast and broadcast: limit multicast and broadcast (1024 kbit/s) Default value: broadcast

VLAN (port-based LAN) register

Configures the use of port-based VLAN.

NOTE

In order for VLAN to be supported, port-based VLAN must be turned off, so that each port can communicate with any other port on the switch.

For each port on a switch, you can configure which other port on the switch received Ethernet frames can be sent to. The table in the VLAN register contains entries that allow the connection between two ports to be set to active or inactive.

	Port				
Port (Ethernet interface to PFF-HM31A)	**Eth 1 (X4233_1)**	**Eth 2 (X4233_2)**	**Eth 3 (X4223)**	**Eth 4**	**COM**
Eth 1 (X4233_1)					
Eth 2 (X4233_2)	Active				
Eth 3 (X4223)	Active	Active			
Eth 4	Active	Active	Active		
COM	Active	Active	Active	Active	
CPU	Active	Active	Active	Active	Active

NOTE

Port Eth 4 has no function.

LLDP register

LLDP (Link Layer Discovery Protocol) periodically sends information about its own unit (e.g. MAC address, unit name, port number) via multicast and receives the same information from neighboring units.

The processor and communications modules support LLDP on ports Eth1, Eth2 and Eth3. Settings for Port Eth4 have no function.

The parameters below specify how the relevant port works.

- Off

  LLDP has been deactivated on this port.
- Send

  LLDP sends LLDP Ethernet frames; received LLDP Ethernet frames are deleted without being processed.
- Receive

  LLDP does not send LLDP Ethernet frames, but received LLDP frames are processed.
- Send/receive

  LLDP sends and processes received LLDP Ethernet frames.

Mirroring register

Configures whether the module duplicates Ethernet packets on a port so these packets can also be read by a unit connected there, e.g. for testing.

The parameters below specify how the relevant port works.

- Off

  This port does not participate in mirroring.
- Egress

  Data leaving this port is duplicated.
- Ingress/egress

  Data arriving at and leaving this port is duplicated.
- Dest. port

  Duplicated data is sent to this port.

9.2.2 Communications module

The communications module (COM) contains the "Module" and "Routings" registers with the same parameters as the processor module. The default IP address default is 192.168.0.100.

9.2.3 Resource configuration

The properties of the resource and the hardware output variables must be configured.

Resource properties

These parameters determine the behavior of the controller during operation and are set in SILworX in the resource's "Properties" dialog.

Parameter/switch	Description	Default value	Settings for safe operation
Name	Name of the resource		Any
System ID [SRS]	System ID of the resource. The system ID value must be different from the default; otherwise the project cannot be run. Setting range: 1 to 65535	60000	Unique value within the network of controllers that may be connected to each other.
Safety time [ms]	Safety time in milliseconds. Setting range: 20 to 22500 ms	600 ms	Application-specific
Watchdog time [ms]	Watchdog time in milliseconds. Setting range: 8 to 5000 ms	200 ms	Application-specific
Main release	The main release can only be set to "ON" if the PES is stopped. **ON:** The following switches/parameters can be changed during operation (= RUN) using the PADT: • *System ID* • *Watchdog time of the resource* • *Safety time* • *Target cycle time* • *Target cycle time mode* • *Autostart* • *Global forcing allowed* • *Global force timeout response* • *Reload function allowed (function available as a unit option)* • *Start allowed* **OFF:** The parameters cannot be changed during operation.	ON	OFF recommended
Autostart	**ON:** If the processor system is connected to the supply voltage, the user program starts automatically. **OFF:** No automatic start after the supply voltage is connected.	OFF	Application-specific
Start allowed	**ON:** Cold start or warm start via PADT allowed in RUN or STOP status. **OFF:** No start allowed.	ON	Application-specific
Loading allowed	**ON:** Downloading the user program allowed. **OFF:** Downloading the user program not allowed.	ON	Application-specific

Parameter/switch	Description	Default value	Settings for safe operation
Reload	**ON:** User program reload function allowed (function available as a unit option). **OFF:** User program reload function not allowed (function available as a unit option). A reload which is already running (function available as a unit option) is not terminated when switched to OFF.	ON	OFF recommended
Global forcing allowed	**ON:** Global forcing allowed for this resource. **OFF:** Global forcing not allowed for this resource.	ON	Application-specific
Global force timeout response	Determines how the resource behaves after the global force timeout has elapsed: • End forcing • Stop resource	End forcing	Application-specific
Max. com. time slice ASYNC [ms]	Maximum value in ms of the time slice that is used for communication during the resource's cycle. Setting range: 2 to 5000 ms	60 ms	Application-specific
Max. duration of configuration connections [ms]	Defines how much time is available during a CPU cycle for process data communication. Setting range: 6 to 5000 ms	6 ms	-
Target cycle time [ms]	Desired or maximum cycle time, see *Target cycle time mode*. The *Target cycle time* cannot exceed the configured watchdog time (6 ms); otherwise the PES rejects it. Setting range: 0 to 7500 ms	0 ms	-
Target cycle time mode	Using the *Target cycle time [ms]*. **Fixed:** The PES adheres to the *Target cycle time* and, if necessary, extends the cycle. This does not apply if the user programs' processing time exceeds the *Target cycle time*. **Fixed/liberal:** Same as fixed, but during the first activation cycle of the reload function (function available as a unit option), the *Target cycle time* is not observed. **Dynamic/liberal:** Same as dynamic, but during the first activation cycle of the reload function (function available as a unit option), the *Target cycle time* is not observed. **Dynamic:** The safety controller adheres to the *Target cycle time* if possible, but executes the cycle in the shortest possible time.	Fixed	-
Minimum configuration version	-	SILworX V4	-
Maximum system bus latency time [µs]	Cannot be used for the safety controller.	0 ms	-
safeethernet CRC	In the current version, the CRC for safeethernet is created with the current algorithm.	Current version	Application-specific

Hardware system variables for creating parameters

These variables are used to change the behavior of the controller during operation if specific statuses occur. These variables are located in the SILworX hardware editor, in the hardware detail view.

Variable	Function	Default setting	Setting for safe operation
Deactivate forcing	Used to prevent and immediately turn off forcing.	FALSE	Application-specific
Blank 2 through Blank 16	No function.	-	-
Emergency off 1 through Emergency off 4	EMERGENCY OFF switch for turning off the controller in the event of malfunctions recognized by the user program.	FALSE	Application-specific
Read-only in RUN	After the controller is started, no further operator action (stop, start, download) is possible via SILworX. Exceptions: Forcing and reload function (function available as a unit option) allowed.	FALSE	Application-specific
Relay contacts 1 through 4	No function.	-	-
Deactivate reload	Prevents loading of the controller via reload function (function available as a unit option).	FALSE	Application-specific
User LEDs 1 to 2	Controls the corresponding LED, if available.	FALSE	Application-specific

Global variables can be assigned to these system variables; the value of the global variables can be changed through physical input or through the user program logic.

Hardware system variables for reading parameters

These system variables can be accessed in the SILworX hardware editor. To do so, select the gray background outside the (yellow) main component display and open the hardware detail view by double-clicking or via the context menu.

Variable	Description	Data type
Number of I/O errors	Number of current I/O errors.	UDINT
Historic number of I/O errors	Total number of I/O errors (counter can be reset).	UDINT
Number of current I/O warnings	Number of current I/O warnings.	UDINT
Historic number of I/O warnings	Total number of I/O warnings (counter can be reset).	UDINT
Number of communication errors	Number of current communication errors.	UDINT
Historic number of communication errors	Total number of communication errors (counter can be reset).	UDINT
Number of communication warnings	Number of current communication warnings.	UDINT
Historic number of communication warnings	Total number of communication warnings (counter can be reset).	UDINT
Number of system errors	Number of current system errors.	UDINT
Historic number of system errors	Total number of system errors (counter can be reset).	UDINT
Number of system warnings	Number of current system warnings.	UDINT
Historic number of system warnings	Total number of system warnings (counter can be reset).	UDINT
Autostart CPU release	**ON:** If the supply voltage is applied, the processor system automatically starts the user program. **OFF:** If the supply voltage is applied, the processor system is switched to STOP status.	BOOL
OS major	Output of the operating system into the processor system.	UINT
OS minor		UINT
CRC	Checksum of the project configuration.	UDINT

Variable	Description	Data type
Date/time [ms portion]	System date and time in s and ms since 01/01/1970.	UDINT
Date/time [sec. portion]		UDINT
Deactivate forcing	**ON:** Forcing is deactivated. **OFF:** Forcing is possible.	BOOL
Forcing is active	**ON:** Global or local forcing is active. **OFF:** Global and local forcing are not active.	BOOL
Force switch status	Status of the force switch. 0xFFFFFFFE: No force switch set 0xFFFFFFFF: At least one force switch set	UDINT
Global forcing started	**ON:** Global forcing is active. **OFF:** Global forcing is not active.	BOOL
Blanks 0 through 16	Reserved	USINT
Blank on17		BOOL
Last I/O warning [ms]	Date and time of the last I/O warning in s and ms since 01/01/1970.	UDINT
Last I/O warning [s]		UDINT
Last communication warning [ms]	Date and time of the last communication warning in s and ms since 01/01/1970.	UDINT
Last communication warning [s]		UDINT
Last system warning [ms]	Date and time of the last system warning in s and ms since 01/01/1970.	UDINT
Last system warning [s]		UDINT
Last I/O error [ms]	Date and time of the last I/O error in s and ms since 01/01/1970.	UDINT
Last I/O error [s]		UDINT
Last communication error [ms]	Date and time of the last communication error in s and ms since 01/01/1970.	UDINT
Last communication error [s]		UDINT
Last system error [ms]	Date and time of the last system error in s and ms since 01/01/1970.	UDINT
Last system error [s]		UDINT
Fan status	0xFF: Not available	BYTE
Major CPU release	Main release switch for the processor system: **ON:** The lower-level release switches can be changed. **OFF:** The lower-level release switches cannot be changed.	BOOL
Read-only in RUN	**ON:** The stop, start and download operator actions are locked. **OFF:** The stop, start and download operator actions are not blocked.	BOOL
Reload release	**ON:** Controller can be loaded via reload function (function available as a unit option). **OFF:** Controller cannot be loaded via reload function (function available as a unit option).	BOOL
Deactivate reloading	**ON:** Loading via reload function is blocked (function available as a unit option). **OFF:** Loading via reload function is allowed (function available as a unit option).	BOOL
Reload cycle	TRUE during the first cycle after a reload function (function available as a unit option); otherwise FALSE.	BOOL
CPU safety time [ms]	Safety time set for the controller in ms.	UDINT

Variable	Description		Data type
Start CPU release	**ON:** Starting the processor system via PADT allowed. **OFF:** Starting the processor system via PADT not allowed.		BOOL
Start cycle	ON during the first cycle after start; otherwise OFF.		BOOL
Power supply status	Bit-coded status of the power supply.		BYTE
	Value	**Status**	
	0x00	Normal	
	0x01	Under-voltage at 24 V supply voltage.	
	0x02	(Under-voltage if battery-powered) unused.	
	0x04	Under-voltage if internally generated voltage is 5 V.	
	0x08	Under-voltage if internally generated voltage is 3.3 V.	
	0x10	Overvoltage if internally generated voltage is 3.3 V.	
System ID	System ID of the controller. Setting range: 1 to 65535		UINT
System tick HIGH	Revolving millisecond counter (64-bit).		UDINT
System tick LOW			UDINT
Temperature status	Bit-coded temperature status of the processor system.		BYTE
	Value	**Status**	
	0x00	Normal temperature	
	0x01	Temperature threshold 1 exceeded	
	0x03	Temperature threshold 2 exceeded	
	0xFF	Not available	
Remaining global force duration [ms]	Time in ms until the global force time limit elapses.		DINT
CPU watchdog time [ms]	Longest allowable duration of a RUN cycle in ms.		UDINT
Cycle time, last [ms]	Current cycle time in ms.		UDINT
Cycle time, max. [ms]	Maximum cycle time in ms.		UDINT
Cycle time, min. [ms]	Minimum cycle time in ms.		UDINT
Cycle time, average [ms]	Average cycle time in ms.		UDINT

User program configuration

The following user program switches and parameters can be set in the user program's "Properties" dialog.

Parameter/switch	Description	Default value	Setting for safe operation
Name	Name of the user program.		Any
Safety integrity level	Safety level: SIL0, SIL3	SIL3	Application-specific
Start allowed	**ON:** Starting the user program via PADT allowed. **OFF:** Starting the user program via PADT not allowed.	ON	Application-specific
Program main release	Release of the change to other user program switches: Only the resource's release switch is relevant.	ON	Application-specific
Autostart	Enabled autostart type: Cold start, warm start, off.	Warm start	Application-specific
Test operation allowed	**ON:** Test operation is allowed for the user program. **OFF:** Test operation is not allowed for the user program.	OFF	Application-specific
Local forcing allowed	**ON:** Forcing at the program level allowed. **OFF:** Forcing at the program level not allowed.	OFF	OFF recommended

Parameter/switch	Description	Default value	Setting for safe operation
Local force timeout response	Behavior of the user program after the force time has elapsed: • End only forcing • Stop program	End only forcing	Application-specific
Reload allowed	**ON:** User program's reload function allowed (function available as a unit option). **OFF:** User program's reload function not allowed (function available as a unit option).	ON	Application-specific
Maximum CPU cycles for program	Maximum number of CPU cycles for which a cycle in the user program can last. A value > 1 is allowed.	1	Application-specific
Max. duration per cycle [µs]	Maximum execution time per processor module cycle for a user program. Setting range: 1 to 7,500,000 µs 0: No limit	0 µs	0 µs
Program ID	ID for identification of the program when displayed in SILworX. Setting range: 1 to 32	1	Application-specific
Watchdog time [ms] (calculated)	Non-changeable monitoring time of the user program, calculated from *Maximum CPU cycles program* and *Resource watchdog time*. **Note:** **If counter inputs are used, you must ensure that the user program's watchdog time is ≤ 5,000 ms.**		-

9.2.4 Configuration of inputs and outputs

The inputs and outputs are configured in the hardware editor by assigning global variables to the system variables for the inputs or outputs.

This is how you find the channels' system variables:

1. View the desired resource in the hardware editor.
2. Open the detail view by double-clicking on the desired input or output module.
3. Open the register with the desired channels in the detail view. The channels' system variables are displayed.

Using digital inputs

The following steps are necessary in order to use the value of a digital input in the user program:

1. Define a global BOOL-type variable.
2. Enter a suitable initial value during definition.
3. Assign the global variable to the input channel value.
4. Program a safety-related error response in the user program using the error code -> *Error code [Byte]*.

The global variable supplies values to the user program.

By assigning global variables to *DI.ErrorCode* and *ModuleErrorCode*, you can program additional error responses in the user program. Details about the error codes are available in the "Parameters and error codes for the inputs and outputs" section.

Using digital outputs

These steps are necessary to write a user program value to a digital output:

1. Define a global BOOL-type variable that contains the value to be output.
2. Enter a suitable initial value during definition.
3. Assign the global variable to the channel value *[BOOL]* -> of the output.
4. Program a safety-related error response in the user program using the error code -> *Error code [Byte]*.

The global variable supplies values to the digital output.

By assigning global variables to *DO.ErrorCode* and *ModuleErrorCode*, you can program additional error responses in the user program.

9.2.5 Generating the resource configuration

Proceed as follows:

1. Select the resource in the structure tree.
2. Click the [Code generation] button in the action bar or select the [Code generation] item from the context menu. The "Code generation" dialog opens.
3. Click [OK] in the "Start code generation" dialog. Another "Start code generation" dialog opens, shows the code generation progress and closes again. A line showing the result of the code generation appears in the log.
4. With the resource still selected, select the [Version comparison] item in the [Extras] menu. The "Version overview" dialog opens. It contains the CRC of the generated code.
5. Click [Export]. An "Archive" dialog is displayed in which you can enter a comment about the project status and the name of the archive file.
6. Generate code again, as described in Steps 2 and 3.
7. With the resource still selected, select the [Version comparison] item in the [Extras] menu. The "Version overview" dialog opens.
8. Click [Import] and in the "Restore " dialog, import the archive file exported in Step 5. The "Version overview" window now contains information about the last generated project status and the imported project status.
9. Click [OK]. The result of the version comparison is displayed in the work area. If "ok" is displayed in the "Comparison of CRCs" column, the code generated for both project statuses is the same and may be used for safety-related operation. Differences are highlighted in red.

The code for resource configuration is now generated.

IMPORTANT

Errors may occur during code generation if a non-secure PC is used.

For safety-related applications, the code generator must generate code twice and the checksums (CRCs) of both generating processes must match. Only then is error-free code ensured.

Additional information is available in the "PFF-HM31A Decentralized Safety Controller for MOVIPRO®" safety manual.

9.2.6 Configuring system ID and connection parameters

Proceed as follows:

1. Select the resource in the structure tree
2. Click the [Online] button in the action bar or select the "Online" item from the context menu.

   The system login dialog window opens.
3. Click [Search]

   The "Search by MAC" dialog window opens.
4. Enter the valid MAC address for the controller (see label on the housing) and click [Search].

   The dialog window shows the values for the IP address, subnet mask and SRS set in the controller.
5. To import the values, click the [Import] button.
6. Log in with the "Administrator" user account
7. Select the [Online]/[Startup]/[Set system ID] menu and assign the desired system ID

   The change is effective immediately, so the connection is terminated.
8. If this has not taken place: Assign the COM & CPU IP address via hardware and transfer the project.
9. To enter additional IP addresses/system IDs, repeat Steps 1 through 6.
10. Loading a program onto the controller

    The change is effective immediately, so the connection is terminated.

Now you can log in with the IP addresses and system IDs configured in the project.

For a system network of several safety controllers, we recommend configuring the individual controllers consecutively before incorporating them into the network. In the safety controller's basic unit configuration, a default value is entered for the IP address. This means the controller can only be assigned using the MAC address of the individual controller.

⚠ WARNING

Danger that the controller being addressed is interchanged. This can cause the wrong user program to be loaded into the safety controller.

Severe or fatal injuries.

When configuring the connection parameters and the system ID set up a point-to-point connection with the corresponding controller.

9.2.7 Loading the resource configuration from the programming device

Before a user program, together with the parameters for connection to the controller (IP address, subnet mask and system ID), can be loaded into the controller, the code for the resource must have been generated. The programming device and resource must also contain valid connection parameters (see "Configuring system ID and connection parameters" chapter).

This is how you load the resource configuration from the programming device:

1. Select resource in the structure tree.
2. Click [Online] in the action bar or select the [Online] entry from the context-sensitive menu.
3. Enter a user group with administrator rights or write access in the "System login" window. The control panel opens in the work area and shows the status of the controller.
4. Select the [Resource download] entry in the [Online] menu. The "Resource download" dialog opens.
5. Confirm the download with "OK" in the dialog. SILworX loads the configuration into the controller.
6. After the user program loads, start with the [Resource cold start] entry in the [Online] menu. After the cold start "System status" and "Program status" change to RUN mode.

The resource configuration has been loaded from the programming device. The "Start", "Stop" and "Load" functions are also available as icons in the icon bar.

9.2.8 Loading the resource configuration from the communication system's flash memory

If data errors in NVRAM cause the watchdog time to be exceeded it may make sense to load the resource configuration from the communication system's flash memory instead of from the programming device.

If the control panel (CP) is no longer accessed, the user program must reconfigure the connection parameters in the controller (see "Configure system ID and connection parameters" chapter).

If the controller enters the STOP/VALID CONFIGURATION status after restarting, the user program can be restarted.

If the controller enters the STOP/INVALID CONFIGURATION status after restarting, the user program must be reloaded into NVRAM.

By using the [Configuration from flash] command, a backup copy of the last executable configuration can be read from the communication system's flash memory and transferred to the processor's NVRAM. Now the user program can be restarted with [Online]/[Resource cold start] without having to download the project.

Here is how you load the resource configuration from the communication system's flash memory:

1. Sign into the desired resource.
2. In the [Online] menu select the [Maintenance/service] submenu and the [Load configuration from flash] entry there.
3. Confirm loading of the configuration in the dialog.

The controller loads the resource configuration from the communication system's flash memory into NVRAM.

9.2.9 Purging the resource configuration from the communication system's flash memory

After temporary hardware errors have occurred the communication system's flash memory may contain remnants of invalid configurations.

The [Purge configuration] command exists for removing these remnants.

Purging the resource configuration:

1. Select the resource in the structure tree.
2. Click [Online] in the action bar or select the [Online] entry from the context-sensitive menu.
3. Enter a user group with administrator rights or write access in the "System login" window. The control panel opens in the work area and shows the status of the controller.
4. Select the [Purge configuration] entry from the [Maintenance/service] submenu of the [Online] menu.
5. Confirm the action with [OK] in the "Purge configuration" dialog.

The configuration in the communication system's flash memory has been purged. Purging of the configuration is only necessary in rare instances. A valid configuration remains untouched during purging.

9.2.10 Set date and time

Proceed as follows:

1. Select the resource in the structure tree.
2. Click [Online] in the action bar or select the [Online] entry from the context-sensitive menu.
3. Enter a user group with administrator rights or write access in the "System login" window. The control panel opens in the work area and shows the status of the controller.
4. Select the [Set date/time] entry from the [Maintenance/service] submenu of the [Online] menu. The "Set date/time" dialog opens.
5. Select one of the options:
   – Use date and time of the programming device.

     This transfers the time and date displayed on the programming device to the controller.
   – User-defined.

     Date and time from the two entry fields are transferred to the controller. Note the listed format when entering the date/time.
6. Clicking [OK] transfers the date and time to the controller. Date and time have been set in the controller.

9.3 User administration with SILworX

SILworX can set up and maintain separate user administrations for each project and for each controller.

9.3.1 User administration for a SILworX project

A PADT user administration can be inserted into each SILworX project; that administration controls access to the project in SILworX.

Without PADT user administratio,n any user can open the project and change all components. If a project has user administration, it can only be opened by a user who has been authenticated. The user can only make changes if he has such rights. The following rights levels exist:

Level	Meaning
Security administrator (Sec Adm)	Can change the user administration: Setting up, deleting, changing of user accounts and user groups and PADT user administration, setting the default user account. In addition, all other SILworX functions are allowed.
Read/Write (R/W)	All SILworX functions except for user administration.
Read-only (RO)	Only read access, no changes, no archiving.

The user administration assigns rights to user groups. The user accounts receive their rights from the user group to which they are assigned.

User group properties:

- The name must be unique within the project and contain 1 – 31 characters.
- A rights level is assigned to a user group.
- Any number of user accounts can be assigned to one user group.
- A project can contain up to 100 user groups.

User account properties:

- The name must be unique within the project and contain 1 – 31 characters.
- A user account is assigned to a user group.
- A project can contain up to 1000 user accounts.
- A user account can be a default user for the project.

9.3.2 User administration for the controller

User administration for a controller (PES user administration) is used to protect the safety controller from unauthorized access. The users and their access rights form part of the project, are defined with SILworX and are loaded into the processor module.

User administration can maintain access rights for a maximum of 10 users per controller. Access rights are saved in the controller and persist even after operating voltage has been turned off.

Each user account consists of a name, password and access right. This information is available for logins once the project has been downloaded to the controller. The users identify themselves with their names and passwords during login into the controller.

Setting up user accounts is not required, but it contributes to safe operation. If a user administration has been defined for a resource, it must contain at least one user with administrator rights.

Default user

If no application-specific user accounts were set up for a resource, the default settings apply.

Default settings:

- Number of users: 1
- User ID: Administrator
- Password: None
- Access right: Administrator

NOTE

Note that if you define your own user accounts, you cannot keep the default settings.

User account parameters

The following parameters must be defined when setting up new user accounts:

Parameter	Description
User name	Name or code with which the user logs into the controller. The user name cannot contain more than 32 characters (recommended: max. 16 characters) and can only contain letters (A to Z, a to z), numbers (0 to 9) and the special characters underline "_" and hyphen "-". It is case-sensitive.
Password	Password associated with the user name; it is required for logging in. The password cannot be longer than 32 characters and can only contain letters (A to Z, a to z), numbers (0 to 9) and the special characters underline "_" and hyphen "-". It is case-sensitive.
Confirm password	Repeat the password to confirm the entry.

Parameter	Description
Access type	The access types define the privileges which a user can have. The following access types can be used: • Read: The user can only read information from the controller, but cannot make any changes. • Read + operator: Same as read, in addition the user can: – Load and start user programs via download – Set processor module to redundant – Reset cycle time and error statistics – Set system time – Override – Restart and reset module – Start system operation for processor modules • Read + write: Same as read + operator, in addition the user can create, transfer, load into the controller and test programs. • Administrator: Same as read + write, in addition the user can: – Load operating systems – Change main release switch – Change SRS – Change IP settings At least one of the users must have administrator rights, otherwise the controller doesn't accept the settings. The administrator can retroactively remove access to a controller from a user by entirely removing the user from the list.

Setting up user accounts

A user with administrator rights has access to all user accounts. When setting up user accounts note the following:

- Ensure that at least one user account with administrator rights has been set up. Define a password for a user account with administrator rights.
- If the administrator has created a user account in user administration and wants to re-edit this account, he must enter the user account's password for authorization purposes.
- Use the SILworX verification to verify user accounts which have been set up.
- The new user accounts apply after the code has been generated and the project has been downloaded to the controller. All previously saved user accounts, e.g. the default setting, become invalid!

9.4 Configuring the communication with SILworX

This chapter describes configuring the communication if the SILworX programming tool is used.

Depending on the application, the following must be configured:

- Ethernet/SafeEthernet
- Default protocols

  For configuration of the default protocols, see "Modbus TCP/UDP" chapter.

9.4.1 Configuring the Ethernet interfaces

Configuration is performed in the communication module's (COM) detail view.

NOTE

SILworX shows the processor system and communication system within a unit or a component as processor module and communication module.

For the safety controller, set the *Speed [Mbit/s]* and *Flow control* parameters in the Ethernet switch settings to "Autoneg". The *ARP Aging Time*, *MAC Learning*, *IP Forwarding*, *Speed [Mbit/s]* and *Flow Control* parameters are explained in detail in the SILworX online help.

NOTE

Swapping a controller with the same IP address:

When a controller for which *ARP Aging Time = 5 minutes* und *MACLearning = conservative* have been set is changed, the communications partner only imports the new MAC address after at least 5 minutes to at most 10 minutes. During that time no communication is possible with the controller that has been swapped.

Parameters for the port settings of the safety controller's integrated Ethernet switch can be set individually. In the "Ethernet switch" register a table entry can be created for each port on the switch.

Port configuration parameters	Explanation
Port	Number of ports as printed on the housing. Each port can only have one configuration. Value range: 1 – n, depending on the resource.
Speed [Mbit/s]	10 Mbit/s: data rate 10 Mbit/s 100 Mbit/s: data rate 100 Mbit/s Autoneg (10/100): baud rate is automatically set Default: autoneg
Flow control	Full duplex: communication in both directions concurrently Half duplex: communication in one direction at a time Autoneg: automatic communications control Default: autoneg
Autoneg also for fixed values	The transmission of speed and flow control properties (Advertising) is also performed if the values for *Speed* and *Flow Control* are fixed. This allows other units whose ports are set to *Autoneg* to recognize the settings of the safety controller's ports.
Limit	Limit incoming multicast and/or broadcast packets. Off: no limit Broadcast: limit broadcast (128 kbit/s) Multicast and broadcast: limit multicast and broadcast (1024 kbit/s) Default: broadcast

The parameters can be changed and entered into the communication system's configuration by double-clicking each table cell. The entries must be re-compiled with the user program and transferred to the controller before they apply to the safety controller's communication.

The properties for the communication system and Ethernet switch can also be changed online via the control panel. These settings apply immediately, but are not imported into the user program.

Details about the configuration of SafeEthernet communication are provided in the "SafeEthernet" chapter.

9.5 Configuring alarms and events

Defining events:

1. Define a global variable for each event. Generally use global variables that have already been defined for the program.
2. Create a new sub-branch "Alarm & events" below the resource, if such a sub-branch doesn't yet exist.
3. Define events in the alarm & events editor
   - Drag the global variable for Boolean or scalar events into the event window.
   - Define the specifics of the events, see the two tables below.

The events have been defined.

The **Boolean event parameters** must be entered in a table which contains the following columns:

Column	Description	Value range
Name	Name of the event definition, must be unique within the resource.	Text, max. 32 characters
Global variable	Name of the global variable assigned (inserted e.g. via drag & drop).	
Data type	Data type of the global variable, cannot be changed.	BOOL
Event source	CPU event: The processor module creates the time stamp. It creates the complete event during each of its cycles. Auto event: same as CPU event Default: auto event	CPU, auto
Alarm if FALSE	Activated: Changing the value of the global variable from TRUE -> FALSE triggers an event. Deactivated: Changing the value of the global variable from FALSE -> TRUE triggers an event. Default: deactivated	Check box activated, deactivated
Alarm text	Text which names the alarm status.	Text
Alarm severity	Severity of the alarm status. Default: 500	0 – 1000
Alarm confirmation successful	Activated: Alarm status must be confirmed by the operator (acknowledgement). Deactivated: Alarm status does not need to be confirmed by the operator. Default: deactivated	Check box activated, deactivated
Return to normal text	Text which names the alarm status.	Text
Return to normal severity	Severity of the normal status. Default: 500	0 – 1000
Return to normal ack required	Normal status must be confirmed by the operator (acknowledgement). Default: deactivated	Check box activated, deactivated

The **scalar event parameters** must be entered in a table which contains the following columns:

Column	Description	Value range
Name	Name of the event definition, must be unique within the resource.	Text, max. 32 characters
Global variable	Name of the global variable assigned (inserted e.g. via drag & drop).	
Data type	Data type of the global variable, cannot be changed.	Depends on the global variable type.
Event source	CPU event: The processor module creates the time stamp. It creates the complete event during each of its cycles. Auto event: same as CPU event Default: auto event	CPU, auto

Column	Description	Value range
HH alarm text	Text which names the alarm status of the highest threshold.	Text
HH alarm value	Highest threshold that triggers an event. Condition: (HH alarm value − Hysteresis) > H alarm value or HH alarm value = H alarm value	Depends on the global variable type
HH alarm severity	Severity of the highest threshold. Default: 500	0 – 1000
HH alarm confirmation required	Activated: Operator must confirm that the highest threshold has been exceeded (acknowledgement). Deactivated: Operator does not need to confirm that the highest threshold has been exceeded. Default: deactivated	Check box activated, deactivated
H alarm text	Text which names the alarm status of the high threshold.	Text
H alarm value	High threshold that triggers an event. Condition: (H alarm value − Hysteresis) > (L alarm value + Hysteresis) or H alarm value = L alarm value	Depends on the global variable type
H alarm severity	Severity of the high threshold. Default: 500	0 – 1000
H alarm confirmation required	Activated: Operator must confirm that the highest threshold has been exceeded (acknowledgement). Deactivated: Operator does not need to confirm that the highest threshold has been exceeded. Default: deactivated	Check box activated, deactivated
Return to normal text	Text which names the alarm status.	Text
Return to normal severity	Severity of the normal status. Default: 500	0 – 1000
Return to normal ack required	Normal status must be confirmed by the operator (acknowledgement) Default: deactivated	Check box activated, deactivated
L alarm text	Text which names the alarm status of the low threshold.	Text
L alarm value	Low threshold that triggers an event. Condition: (L alarm value + Hysteresis) < (H alarm value − Hysteresis) or L alarm value = H alarm value	Depends on the global variable type
L alarm severity	Severity of the low threshold. Default: 500	0 – 1000
L alarm confirmation required	Activated: Operator must confirm dropping below the low threshold (acknowledgement). Deactivated: Operator does not need to confirm dropping below the low threshold. Default: deactivated	Check box activated, deactivated
LL alarm text	Text which names the alarm status of the lowest threshold.	Text
LL alarm value	Lowest threshold that triggers an event. Condition: (LL alarm value + Hysteresis) < (L alarm value) or LL alarm value = L alarm value	Depends on the global variable type
LL alarm severity	Severity of the lowest threshold. Default: 500	0 – 1000
LL alarm confirmation required	Activated: Operator must confirm dropping below the lowest threshold (acknowledgement). Deactivated: Operator does not need to confirm dropping below the lowest threshold. Default: deactivated	Check box activated, deactivated
Alarm hysteresis	Hysteresis avoids continuous creation of many events if the process value frequently fluctuates around a threshold.	Depends on the global variable type

IMPORTANT!

Incorrect event creation due to errors in parameter set possible!

Setting the *L Alarm Value* and *H Alarm Value* to the same value may cause event creation to behave undesirably because no normal range exists in that case.

Ensure therefore that *L Alarm Value* and *H Alarm Value* are set to different values.

9.6 Working with the user program

By using the programming device the user can influence how his program functions on the controller as follows:

9.6.1 Setting parameters and switches

While a user program's project is in the process of being designed, the parameters and switches are set offline and are loaded into the controller with the code-generated program. Parameters and switches can also be set in STOP and RUN status if the *Main Release* switch has been set. Only the elements in NVRAM can be changed, all others are set during loading.

9.6.2 Starting the program from STOP/VALID CONFIGURATION

Starting the program corresponds to switching the controller from the STOP/VALID CONFIGURATION operating mode to the RUN operating mode. The program, too, switches into RUN mode. The program switches into test mode, if the test mode has been activated during the start. According to IEC 61131, a cold or warm start is also possible in addition to starting in test mode.

NOTE

Starting the program is only possible if the *Start/Restart* switch has been set to allowed.

9.6.3 Program restart after error

If the program switches to STOP/INVALID CONFIGURATION, e.g. due to unauthorized access to operating system sections, it restarts. If the program returns to the STOP/INVALID CONFIGURATION status within approx. one minute after restart, it remains in that status. It can then be restarted via the Start button on the control panel. After the restart, the operating system verifies the entire program.

9.6.4 Stopping the program

If a user program is stopped, the controller switches from the RUN operating mode to the STOP/VALID CONFIGURATION mode.

9.6.5 Program test mode

The test mode is started via the control panel in the [Test mode]/[Test mode with hot start] (or cold start/warm start) menu. By using the *Cycle Step* command, a single step is activated each time (single logic pass).

Behavior of variables/signal values in test mode:

The selection cold start, warm start or hot start sets which variable values are used for the first run through the test mode.

- Cold start: All variables/signals receive their initial value.
- Warm start: Retain signals keep their value, others are set to their initial values.
- Hot start: All variables/signals keep their current value.

Next the user program can be started in single step mode by using the *Cycle Step* command. All current values are retained for the next cycle (frozen status).

⚠ WARNING!

Actuators are in non-secure status.

Severe or fatal injuries.

Do not use the test mode function in safety-related operation!

9.6.6 Online test

The online test function permits insertion of online test fields (OLT fields) into the programming logic; during controller operation these fields are used to display and override signals/variables.

If the *Online Test Allowed* switch is turned on, values for signals/variables can be manually entered in the corresponding OLT fields – and thereby used for overriding – while the program is running. The override value, however, only applies until it is overwritten in the program logic.

If the *Online Test Allowed* switch is turned off, the values for signals/variables can only be displayed in OLT fields, but cannot be changed.

Additional information about using OLT fields is available in the programming tool's online help under the keyword "OLT field".

10 Operation

This chapter describes operation and diagnostics while the controller is in operation.

10.1 Operating

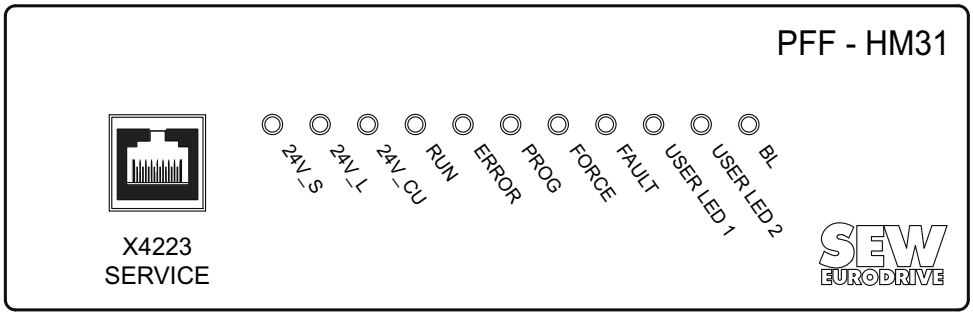
Operating the controller is not necessary during regular operation. Only when problems occur may intervention with the programming be necessary.

10.2 Diagnostics

A rough initial diagnosis can be performed by using the light-emitting diode displays. A detailed analysis of the operating or error status can be performed by using the diagnostic history. That history can be displayed with the programming device.

10.2.1 LED display

The system LEDs are located on the service unit and show the status of the fieldbus and the unit. There are also 2 user LEDs which can be configured by the user as required:



4867138571

The following table shows the status and the meaning of the LED:

Designation	Status LED	Meaning
BL	Flashing red	• BL (boot loader) defect or hardware error. • External process data communications error • Duplicate IP address detected.[1)]
	Off	None of the events described have occurred.
USER LED 2 USER LED 1	Red light	Coding: 1
	Flashing red	Coding: 2
	Off	Coding: 0 or 3...255
FAULT	Yellow light / flashing yellow[2)]	• The new operating system is corrupted (after downloading) • Error while loading a new operating system • Incorrect configuration loaded. • One or more I/O errors have occurred. • Duplicate IP address detected[1)].
	Off	None of the events described have occurred.
FORCE	Yellow light	Overriding prepared: • The override switch of a variable has been set • and the main override switch has not yet been enabled. • The unit is set to the status RUN or STOP.
	Flashing yellow	• Overriding is active: At least one local or global variable has assumed its override value. • Duplicate IP address detected[1)].
	Off	None of the events described have occurred.

Designation	Status LED	Meaning
PROG	Yellow light	• Loading the unit with a new configuration. • Loading a new operating system. • Changes to the WDT or ETT. • Check for duplicate IP address. • Changes to the SRS.
	Flashing yellow	• Executing reload function (function is available as a unit option) • Duplicate IP address detected[1].
	Off	None of the events described have occurred.
ERROR	Red light / flashing red[2]	• The unit status is set to FAULT STOP: Internal errors detected by self-tests, e.g. hardware error, software error or voltage supply error. Remedy: The processor system can only be restarted by a command from the PADT (reboot). • Protocols/functions that have not been activated are used (warning). • Error while loading the operating system
	Off	None of the events described have occurred.
RUN	Lights up green	• Unit status set to RUN, normal operation • A loaded user program is running
	Flashing green	• Unit status set to STOP • Loading a new operating system
	Off	The unit status is not set to RUN or STOP.
24V_CU	Lights up green	24 V is present between X1541.1 and X1541.2.
24V_L	Lights up green	24 V is present between X1541.3 and X1541.4.
24V_S	Lights up green	24 V is present between X2312.1 and X2312.3.

1) If both LEDs are flashing at once: PROG, FORCE, FAULT and BL

2) The "lit" status signals a warning and "flashing" signals an alarm.

Whenever the voltage supply is connected, all of the LEDs light up for a short time whilst they are tested.

User LEDs

The two freely configurable user LEDs are controlled using system variables. Global variables of the USINT data type must be assigned to the corresponding system variables in order to do so.

10.2.2 Diagnostic history

The diagnostic history records the various processor and communication system statuses and stores them in non-volatile memory. Here long-term and short-term diagnostics are distinguished according to the table below.

	CPU	COM
Entries in long-term diagnostics	700	300
Entries in short-term diagnostics	700	700

The processor system's long-term diagnostics include the following events:

- Reboot
- Change in operating mode
  (INIT, RUN, STOP/VALID CONFIGURATION, STOP/INVALID CONFIGURATION)
- Change in program operating mode
  (START, RUN, ERROR, TEST MODE)
- Loading/deleting a configuration
- Setting and resetting switches
- Error in the processor system
- Loading an operating system
- Overriding (setting and resetting of the override switch allowed)
- Diagnosis of power supply and temperature

The communication system's long-term diagnostics include the following events:

- Rebooting the communication system
- Change in operating mode (INIT, RUN, STOP/VALID CONFIGURATION, STOP/ INVALID CONFIGURATION)
- Users log in
- Loading an operating system

If the long-term diagnostic memory is full, all data older than three days is deleted and new entries can be accepted. If all data is less than three days old, no new data can be saved and such data is lost. An entry in long-term diagnostics shows that data could not be saved.

The processor system's short-term diagnostics include the following events:

- Diagnostic of the processor system (setting the override switches and override values)
- Diagnostic of the user program (cycle operation)
- Diagnostic of the communication
- Diagnostic of the power supply and temperature

The communication system's short-term diagnostics include the following events:

- SafeEthernet-related events
- Start/stop while writing to flash memory
- Errors which can occur when a configuration is loaded from flash memory
- Diverging time synchronization between communication system and processor system

Errors in setting parameters for inputs and outputs may not be recognized during code generation. If an error occurred when parameters were set, the diagnostics' response window displays the message INVALID CONFIG with the source of the error and an error code. This message helps to analyze errors while setting parameters for input and output.

If the short-term diagnostic memory is full, the oldest entries are removed to make room for new entries. There is no display when old entries are deleted.

Recording of diagnostic data is not safety-related. The data recorded in chronological order can be read for analysis by using the programming tool. Reading does not delete the data in the controller. The programming tool can save the content of the diagnostics window.

10.2.3 Diagnostics in SILworX

The diagnostics are accessed via the online view in the SILworX hardware editor.

Proceed as follows to open the diagnostics:

1. Mark the "Hardware" branch below the desired resource.
2. Click [Online] in the context-sensitive menu or in the action bar. The system login window opens.
3. Select or enter the following information in the system login window:
   – IP address of the controller
   – User and password

   The hardware editor's online view opens.
4. Select the desired module in the online view; usually this is the processor or communication module.
5. Select the [Diagnostics] item from the context-sensitive menu or the [Online] menu.

The diagnostics for the corresponding module open.

If the controller is running, messages about statuses of the processor system, the communication system and the I/O components during specific, adjustable intervals, are displayed.

10.3 Parameters and error codes for inputs and outputs

The following overviews list readable and adjustable system parameters for inputs and outputs, including error codes. The error codes can be read within the user program via corresponding variables assigned in the logic. The error codes can also be displayed in SILworX.

10.3.1 Digital inputs PFF-HM31A

The tables below contain the status and parameters of the input module (DI 26) in the same order as in the hardware editor.

Register module

The register module contains the following system parameters:

System parameter	Data type	R/W	Description	
DI NumberClock-FeedChannels	USINT	W	Number of clock outputs (feed outputs)	
			Coding	**Description**
			0	No clock output intended for LF/LB[1)] recognition
			1	Clock output 1 intended for LF/LB[1)] recognition
			2	Clock outputs 1 and 2 intended for LF/LB[1)] recognition
			...	...
			6	Clock outputs 1 through 6 intended for LF/LB[1)] recognition
			Clock outputs cannot be used as safety-related outputs!	
DI Feed [01]	BOOL	W	Control of supply output SS0. TRUE: Feed is turned on FALSE: Feed is not turned on	
DI OptionSlot Clock feed component	UDINT	W	Option slot for the clock feed component (LF/LB[1)] recognition), set value to "2".	
DI ClockDelay [µs]	UINT	W	Waiting time for line control (fault/short circuit recognition).	
DI.ErrorCode	WORD	R	Error codes for all digital inputs.	
			Coding	**Description**
			0x0001	Errors in the component
			0x0002	ETT test of the test pattern had errors
			0x2000	Setting parameters for LF/LB[1)] recognition had errors

System parameter	Data type	R/W	Description	
DI.ErrorCode Feed	WORD	R	Component error code for supply output SS0	
			Coding	**Description**
			0x0001	Errors in the component
DI[01].ErrorCode Feed	BYTE	R	Channel error code for supply output SS0.	
			Coding	**Description**
			0x01	Error DI Feed Unit
			0x02	Feed turned off due to overcurrent
			0x04	Error when reading back the feed
DO.ErrorCode	WORD	R	Error codes for all clock outputs	
			Coding	**Description**
			0x0001	Errors in the component
ModuleErrorCode	WORD	R	Error codes for the module	
			Coding	**Description**
			0x0000	I/O processing, if necessary with errors, see additional error codes
			0x0001	No I/O processing (CPU not in RUN)
			0x0002	No I/O processing during boot-up test
			0x0004	Manufacturer's interface in operation
			0x0010	No I/O processing: incorrect parameters set
			0x0020	No I/O processing: error rate exceeded
			0x0040/ 0x0080	No I/O processing: configured module not plugged in
ModuleSRS	UDINT	R	Option slot number (system – rack – slot)	
ModuleType	UINT	R	Module type, target value : 0x001A [26_{dec}]	

1) LF = line fault / LB = line break (line control)

Register DI26: DI channels

The Register DI 26: DI channels includes the following system parameters:

System parameter	Data type	R/W	Description	
Channel No.	-	R	Channel number, fixed	
-> Error code [BYTE]	BYTE	R	Error codes of the digital input channels	
			Coding	**Description**
			0x01	Error in the digital input module
			0x10	Line fault in channel
			0x80	Disruption between clock output TO and digital input DI, e.g. • line break • open switch • L+ undervoltage (+24 V_S)
-> Value [BOOL]	BOOL	R	Input value of digital inputs. 0 = Input not controlled 1 = Input controlled	
Clock feed channel [USINT ->	USINT	W	Source channel for clock feed	
			Coding	**Description**
			0	Input channel
			1	Clock from 1st TO channel
			2	Clock from 2nd TO channel
			...	...
			6	Clock from 6th TO channel

Register DI26: DO channels

The Register DI 26: DO channels includes the following system parameters:

System parameter	Data type	R/W	Description	
Channel No.	-	R	Channel number, fixed	
-> Error code [BYTE]	BYTE	R	Error codes of the digital outputs	
			Coding	**Description**
			0x01	Error in the digital output module or in the component
-> Value [BOOL]	BOOL	W	Output value of the DO channels	
			Coding	**Description**
			0	Output without current
			1	Output controlled

10.3.2 Digital outputs PFF-HM31A

The tables below contain the status and parameters of the output module (DO 8) in the same order as in the hardware editor.

Register module

The register module contains the following system parameters:

System parameter	Data type	R/W	Description	
DO.ErrorCode	WORD	R	Error codes for all digital outputs.	
			Coding	**Description**
			0x0001	Errors in the component
			0x0002	MEZ test of the emergency stop results in an error
			0x0004	MEZ test of the auxiliary power supply results in an error
			0x0008	ETT test of the test pattern had errors
			0x0010	MEZ test of the output switch test pattern had errors
			0x0020	MEZ test of the output switch test pattern (disconnect test of the outputs) had errors
			0x0040	MEZ test active disconnect via WD had errors
			0x0400	ETT test: 1. Temperature threshold exceeded
			0x0800	ETT test: 2. Temperature threshold exceeded
			0x4000	Setting parameters for 2-pole monitoring had errors
Activation delay	UINT	W	Activation delay for 2-pole tests, due to line capacities, inductive and capacitive load. Value range: 0 – 30 ms	
ModuleErrorCode	WORD	R	Error codes for the module	
			Coding	**Description**
			0x0000	I/O processing, if necessary with errors, see additional error codes
			0x0001	No I/O processing (CPU not in RUN)
			0x0002	No I/O processing during boot-up test
			0x0004	Manufacturer's interface in operation
			0x0010	No I/O processing: incorrect parameters set
			0x0020	No I/O processing: error rate exceeded
			0x0040/ 0x0080	No I/O processing: configured module not plugged in
ModuleSRS	UDINT	R	Option slot number (system – rack – slot)	
ModuleType	UINT	R	Module type, target value : 0x0029 [41_{dec}]	

Register DO 8: Channels

The Register DO 8: Channels includes the following system parameters:

System parameter	Data type	R/W	Description	
Channel No.	-	R	Channel number, fixed.	
-> + Error code [BYTE]	WORD	R	Error codes of the digital outputs.	
			Coding	**Description**
			0x0001	Error in the digital output module.
			0x0002	Output turned off due to overvoltage.
			0x0004	Error when reading back control of digital outputs.
			0x0008	Error when reading back the status of digital outputs.
			0x0020	External line fault or EMC protection fault causes an error (L+ fault (24V_L)).
			0x0040	External line fault or EMC protection fault causes an error (L− fault (0V24)).
			0x0080	Channel has been turned off due to error in associated DO channel.
			0x0100	Test output fault against L+ (24V_L) not performed due to changes in target values or undervoltage.
			0x0200	Test output fault against L− (0V24) not performed due to changes in target values or undervoltage.
			0x0400	All channels have been turned off, overall voltage exceeded.
			0x0800	ETT test: Monitoring of auxiliary power supply 1: Undervoltage.
-> − Error code [BYTE]	WORD	R	See -> + *Error Code[BYTE]*	
-> Value [BOOL]	BOOL	W	Output value of the DO channels. 0 = Output without current 1 = Output controlled	
2-pole turned off [BOOL] ->	BOOL	W	Parameter sets whether channel is used in 2-pole mode. 0 = Channel used in 2-pole mode 1 = Channel used in 1-pole mode	

10.3.3 Counter PFF-HM31A

The tables below contain the status and parameters of the counter module (HSC 2) in the same order as in the hardware editor.

Register module

The register module contains the following system parameters:

System parameter	Data type	R/W	Description	
ModuleErrorCode	WORD	R	Error codes for the module.	
			Coding	**Description**
			0x0000	I/O processing, if necessary with errors, see additional error codes.
			0x0001	No I/O processing (CPU not in RUN).
			0x0002	No I/O processing during boot-up test.
			0x0004	Manufacturer's interface in operation.
			0x0010	No I/O processing: incorrect parameters set.
			0x0020	No I/O processing: error rate exceeded.
			0x0040/ 0x0080	No I/O processing: configured module not plugged in.
ModuleSRS	UDINT	R	Option slot number (system – rack – slot)	
ModuleType	UINT	R	Module type, target value : 0x0003 [3_{dec}]	

System parameter	Data type	R/W	Description	
Counter.ErrorCode	WORD	R	Error codes for the counter module.	
			Coding	**Description**
			0x0001	Errors in the component.
			0x0002	Error when comparing base time.
			0x0004	Address error when reading the base time.
			0x0008	Parameter for base time has errors.
			0x0010	Address error when reading the count.
			0x0020	Configured counter parameters have been corrupted.
			0x0040	Address error when reading the gray code.
			0x0080	ETT test of the test pattern had errors.
			0x0100	ETT test error when checking the coefficient
			0x0200	Error during setting of initial parameters for the component.

Register HSC 2: Channels

The Register HSC 2: Channels includes the following system parameters:

System parameter	Data type	R/W	Description	
Counter[0x].5/24V mode	BOOL	R/W	Counter input 5 V or 24 V. TRUE: 24 V FALSE: 5 V	
Counter [0x]Autom.Detect-ingDirectionofRota-tion	BOOL	R/W	Automatically detect direction of rotation. TRUE: Automatically detect direction of rotation ON FALSE: Manually set direction of rotation	
Counter[0x].Error-Code	BYTE	R	Error codes of the counter channels.	
			Coding	**Description**
			0x01	Error in the counter module
			0x02	Error when comparing counts.
			0x08	Error when setting the parameters (reset).
Counter[0x].Gray-Code	BOOL	R/W	Decoder/ImpulseOperation. TRUE: Gray code decoder FALSE: Impulse operation **Decoder operation not permitted!**	
Counter[0x].Blank1	BOOL	R/W	No function	
Counter[0x].Blank2	BOOL	R/W		
Counter[0x].Blank3	BOOL	R/W		
Counter[0x].Reset	BOOL	R/W	Reset of the counter channel (only if *Counter[0x].Autom. DetectingDirectionofRotation = FALSE*) TRUE: No reset FALSE: Reset	
Counter[0x].Direc-tion	BOOL	R/W	Counting direction of the counter (only if *Counter[0x].Autom. DetectingDirectionofRotation = FALSE*) TRUE: Down (decrementing) FALSE: Up (incrementing):	
Counter[0x].Value	UDINT	R	Count of the counters: 24 Bit for impulse counter.	
Counter[0x]. ValueOverflow	BOOL	R	Counter overflow display TRUE: Overflow since the last cycle (only if *Counter[0x].Autom. DetectingDirectionofRotation = FALSE*) FALSE: No overflow since the last cycle	
Counter[0x].Time-Stamp	UDINT	R	Time stamp for *Counter[0x].Value*, 24 Bit, time resolution 1 µs.	
Counter[0x]. TimeOverflow	BOOL	R	Overflow display for the counter's time stamp TRUE: 24 Bit overflow since last cycle FALSE: No overflow since the last cycle	

11 Servicing

Servicing of the safety controller is limited to the following:

- Removing faults
- Loading operating systems

11.1 Fault information

Most faults in the processor system (CPU) cause the whole control unit to shut down and are shown by the "ERROR" status LED.

You can delete the display by executing the "Reboot resource" command in the [Extra] menu on the SILworX control panel. The control unit is booted up and restarted. The system automatically detects faults in the input and output channels during operation. These faults are signaled by the "FAULT" status LED on the top of the unit.

The PADT (SILworX) also offers the option to read detected errors using the diagnostics tool unless there is also a communication error.

- Before replacing a control unit, check for an external line fault and ensure that the corresponding sensor/actuator is in working order.

11.2 Loading operating systems

Different operating systems are used for the processor system and communication system. These are stored on rewritable flash memories and can be replaced if necessary.

⚠ WARNING!

Interruption of safety-related operation caused by new operating systems being loaded by the programming tool.

Severe or fatal injuries!

- The control unit status must be set to STOP to enable the programming tool to load new operating systems.
- The operator must ensure that the safety of the system is guaranteed during this time, e.g. by taking the necessary organizational measures.

NOTES

- The programming tool prevents operating systems from being loaded when the status is set to RUN and signals this accordingly.
- If the loading process is interrupted or incorrectly terminated, the control unit will no longer function. You can, however, reload an operating system.

You must load the operating system for the processor system (CPU operating system) before you load the operating system for the communication system (COM operating system). In order for a new operating system to be loaded, it must be stored in a directory which can be accessed using the programming tool.

11.2.1 Loading operating systems with SILworX

Proceed as follows to load a new operating system:

1. Set the control unit status to STOP unless this has already taken place.
2. Open the online view for the hardware and log into the control unit with administrator rights.
3. Right click the module to be loaded (processor module or communication module).
4. Click [Maintenance/Service] / [Load operating system module] in the open context-sensitive menu.
5. Select the type of operating system to be loaded in the "Load operating system module" dialog.
6. In the open file selection window, select the file containing the operating system to be loaded and click [Open].

SILworX will load the new operating system into the control unit.

12 Appendix

12.1 Glossary

Term	Description
ARP	Address resolution protocol (network protocol for assigning network addresses to hardware addresses)
BL	Boot loader
BS	Operating system
COE	CANopen software module
COM	Communication module
CRC	Cyclic redundant check (checksum)
CUT	Com-user task
DC-24V	The safety controller has the following DC 24 V input voltage potentials: 24V_CU: DC 24V input – control unit 24V_L: DC 24V input – load 24V_S: DC 24V input – sensor supply Reference potential: 0V24
DCS	Distributed control system (process guidance system)
DI	Digital input (binary input)
DO	Digital output (binary output)
EMC	Electromagnetic compatibility:
EN	European standard
ESD	Electrostatic discharge
ESPE	Electro-sensitive protective equipment
ETT	Error tolerance time
FB	Fieldbus interface of the control unit
FBD	Function block diagram
FIFO	First in first out (data memory)
FTA	Field termination assembly
ICMP	Internet control message protocol (network protocol for status and error messages)
IEC	International Electrotechnical Commission Standards
IF	Interface
MAC address	Hardware address of a network connection (Media Access Control)
Non-impacting	Two input fuses are connected to the same source (e.g. transmitter). If one of the input fuses does not interfere with the signal from the other input fuse, it is referred to as 'non-impacting'.
NVRam	Non volatile random access memory
PADT	Programming and debugging tool (in accordance with IEC 61131-3), PC with SILworX
PE	Protective earth
PELV	Protective extra low voltage
PES	Programmable electronic system
POU	Program organizational units (in accordance with IEC 61131-1)
PFD	Probability of failure on demand (probability of failure when requesting a safety function)
PFF-HM31A	Safety controller
PFH	Probability of failure per hour
R	Read (System variable/signal sends the value to the user program, for example)
R/W	Read/Write (column header for type of system variable / signal)
SB	System bus (module)
SELV	Safety extra low voltage
SFF	Safe failure fraction
SIL	Safety integrity level (according to IEC 61508)
SILworX	Programming tool for safety controller PFF-HM31A
SNTP	Simple network time protocol (RFC 1769)

Term	Description
S.R.S	System.Rack.Slot (addressing a module)
SW	Software
S&R	Send and receive; related to TCP protocol
TMO	Timeout
W	Write (system variable/signal is supplied with a value, e.g. from the user program)
Watchdog (WD)	Time monitoring for modules or programs A fault stop will occur in the module or program if the watchdog time is exceeded.
WDT	Watchdog time

Index




SEW-EURODRIVE
Driving the world

SEW-EURODRIVE GmbH & Co KG
P.O. Box 3023
D-76642 Bruchsal/Germany
Phone +49 7251 75-0
Fax +49 7251 75-1970
sew@sew-eurodrive.com

➞ **www.sew-eurodrive.com**