# HIMatrix M45

## Safety-Related Controller

## System Manual M45

HIMA Paul Hildebrandt GmbH
Industrial Automation

Rev. 1.01

HI 800 651 E

For further information, refer to the HIMA DVD and our website http://www.hima.de and http://www.hima.com.



## Contact

HIMA contact details:

HIMA Paul Hildebrandt GmbH

P.O. Box 1261

68777 Brühl, Germany

Phone: +49 6202 709-0

Fax: +49 6202 709-107

E-mail: info@hima.com

| Revision index | Revisions | Type of change | |
|---|---|---|---|
| | | technical | editorial |
| 1.00 | First issue | | |
| 1.01 | Revision, changed chapters: Extending the System Bus, Mounting and Removing Modules and Sockets | X | X |
| | | | |
| | | | |

# Table of Contents

# 1 Introduction

The modular HIMatrix M45 system described in this manual is safety-related and can be used for various purposes. The following conditions must be met to safely install and start up the HIMatrix M45 automation devices, and to ensure safety during their operation and maintenance:

- Knowledge of regulations.
- Proper technical implementation of the safety instructions detailed in this manual performed by qualified personnel.

HIMA shall not be held liable for severe personal injuries, damage to property or the environment caused by any of the following: unqualified personnel working on or with the devices, de-activation or bypassing of safety functions, or failure to comply with the instructions detailed in this manual (resulting in faults or impaired safety functionality).

HIMatrix M45 automation devices have been developed, manufactured and tested in compliance with the pertinent safety standards and regulations. They may only be used for the intended applications under the specified environmental conditions and only in connection with approved external devices.

## 1.1 Structure and Use of the Document

This system manual is composed of the following chapters:

| | |
|---|---|
| Safety | Information on how to safely use the HIMatrix M45 system.<br>Allowed applications and environmental requirements for operating the HIMatrix M45 systems. |
| Product description | Basic structure of the HIMatrix M45 system. |
| Communication | Brief description of the HIMatrix M45 modular systems' communication among each other and with other systems. Detailed information can be found in the communication manuals. |
| Operating system | Functions of the operating systems. |
| User program | Basic information on the user program. |
| Start-up, operation, maintenance, decommissioning, transport, disposal | Phases of a HIMatrix M45 system's lifecycle. |
| Appendix | ▪ Glossary<br>▪ Index of tables and index of figures<br>▪ Declaration of conformity<br>▪ Index |

Additionally, the following documents must be taken into account:

| Name | Content | Document number |
|---|---|---|
| HIMatrix M45 Safety Manual | Safety functions of the HIMatrix M45 system | HI 800 653 E |
| SILworX Communication Manual | Description of the communication protocols, ComUserTask and their configuration in SILworX | HI 801 101 E |
| HIMatrix M45 module manuals | Description of the HIMatrix M45 modules | |
| SILworX Online Help (OLH) | Instructions on how to use SILworX | - |
| SILworX First Steps | Introduction to SILworX using the HIMax system as an example | HI 801 103 E |

Table 1: Additional Relevant Documents

The latest manuals can be downloaded from the HIMA website at www.hima.com. The revision index on the footer can be used to compare the current version of existing manuals with the Internet edition.

## 1.2 Target Audience

This document addresses system planners, configuration engineers, programmers of automation devices and personnel authorized to implement, operate and maintain the modules and systems. Specialized knowledge of safety-related automation systems is required.

## 1.3 Formatting Conventions

To ensure improved readability and comprehensibility, the following fonts are used in this document:

| | |
|---|---|
| **Bold** | To highlight important parts.<br>Names of buttons, menu functions and tabs that can be clicked and used in the programming tool. |
| *Italics* | For parameters and system variables. |
| `Courier` | Literal user inputs. |
| RUN | Operating states are designated by capitals. |
| Chapter 1.2.3 | Cross-references are hyperlinks even if they are not particularly marked. When the cursor hovers over a hyperlink, it changes its shape. Click the hyperlink to jump to the corresponding position. |

Safety notes and operating tips are particularly marked.

### 1.3.1 Safety Notes

The safety notes are represented as described below.
These notes must absolutely be observed to reduce the risk to a minimum. The content is structured as follows:

- Signal word: warning, caution, notice
- Type and source of risk
- Consequences arising from non-observance
- Risk prevention

**SIGNAL WORD**

**Type and source of risk!**

**Consequences arising from non-observance**

**Risk prevention**

The signal words have the following meanings:

- Warning indicates hazardous situation which, if not avoided, could result in death or serious injury.
- Caution indicates hazardous situation which, if not avoided, could result in minor or modest injury.
- Notice indicates a hazardous situation which, if not avoided, could result in property damage.

**NOTE**

**Type and source of damage!**

**Damage prevention**

### 1.3.2 Operating Tips

Additional information is structured as presented in the following example:

i The text corresponding to the additional information is located here.

Useful tips and tricks appear as follows:

**TIP** The tip text is located here.

## 1.4 Service and Training

Deadlines and the extent of actions for commissioning, testing and modifying controller systems can be agreed with the service department.

HIMA holds training, usually in-house, for software programs and the hardware of the controllers. Additionally, customer training can be offered on-site.

Refer to the HIMA website at `www.hima.com` for the current training program and dates. Offers for specialized, on-site training can also be provided upon request.

# 2 Safety

All safety information, notes and instructions specified in this document must be strictly observed. The product may only be used if all guidelines and safety instructions are adhered to.

The product is operated with SELV or PELV. No imminent risk results from the product itself. The use in Ex-Zone is permitted if additional measures are taken.

## 2.1 Intended Use

This chapter describes the conditions for using HIMatrix M45 systems.

### 2.1.1 Scope

The safety-related HIMatrix controllers can be used in applications up to SIL 3 in accordance with IEC 61508.

The HIMatrix systems are certified for use in process controllers, protective systems, burner controllers, and machine controllers.

#### 2.1.1.1 Application in Accordance with the De-Energize-to-Trip Principle

The automation devices have been designed in accordance with the de-energize-to-trip principle.

A system that operates in accordance with the de-energize-to-trip principle adopts the de-energized state if a fault occurs.

#### 2.1.1.2 Use in Fire Alarm Systems

HIMatrix systems are suitable for use in fire alarm systems in accordance with DIN EN 54-2 and NFPA 72. For these fire alarm systems, the following is required:

- To contain the risks, these systems must be able to adopt an active state on demand.
- Open-circuit and short-circuit detection.

The environmental requirements have to be observed!

For details, see the safety manual of the HIMatrix F systems (HI 800 023 E).

## 2.2 Environmental Requirements

| Requirement type | |
|---|---|
| Protection class | Protection class III in accordance with IEC/EN 61131-2 |
| Ambient temperature | 0...+60 °C |
| Storage temperature | -40...+85 °C |
| Pollution | Pollution degree II in accordance with IEC/EN 61131-2 |
| Altitude | < 2000 m |
| Housing | Standard: IP20 |
| Supply voltage | 24 VDC |

Table 2: Environmental Requirements

All the environmental requirements specified in this manual must be observed when operating the HIMatrix system.

### 2.2.1 Test Conditions

The devices have been tested to ensure compliance with the following standards for EMC, climatic and environmental requirements:

| Standard | Content |
|---|---|
| IEC/EN 61131-2: 2007 | Programmable controllers, Part 2<br>Equipment requirements and tests |
| IEC/EN 61131-6: 2012 | Programmable controllers, Part 6<br>Functional Safety |
| IEC/EN 61000-6-2: 2005 | EMC<br>Generic standards, Parts 6-2<br>Immunity for industrial environments |
| IEC/EN 61000-6-4: 2007 + A1:2011 | Electromagnetic compatibility (EMC)<br>Generic emission standard, industrial environments |

Table 3: Standards for EMC, Climatic and Environmental Requirements

When using the safety-related HIMatrix control systems, the following general requirements must be met:

| Requirement type | Requirement content |
|---|---|
| Protection class | Protection class III in accordance with IEC/EN 61131-2 |
| Pollution | Pollution degree II in accordance with IEC/EN 61131-2 |
| Altitude | < 2000 m |
| IP class<br>Requirements of the application | Standard: IP20<br>If required by the relevant application standards (e.g., EN 60204, EN 13849), the HIMatrix system must be installed in an enclosure of the specified protection class (e.g., IP54). |

Table 4: General Requirements

### 2.2.1.1 Climatic Requirements

The following table lists the most important tests and limits for climatic requirements:

| Standard | Climatic tests |
|---|---|
| IEC/EN 61131-2 | Operating temperature: 0...+60 °C<br>(test limits: -10...+70 °C) |
| | Storage temperature: -40...+85 °C |
| | Dry heat and cold resistance tests:<br>+70 °C / -40 °C, 16 h, +85 °C, 1 h<br>Power supply not connected |
| | Temperature changes, withstand test:<br>Fast temperature changes: -40 °C / +70 °C power supply not connected |
| | Immunity test:<br>Slow temperature changes: -10 °C / +70 °C power supply not connected |
| | Cyclic damp-heat withstand tests:<br>+25 °C / +55 °C, 95 % relative humidity,<br>Power supply not connected |
| EN 54-2 | Damp-heat<br>93 % relative humidity, 40 °C, 4 days in operation<br>93 % relative humidity, 40 °C, 21 days, power supply not connected |

Table 5: Climatic Requirements

Operating requirements other than those specified in this document are described in the device-specific or module-specific manuals.

### 2.2.1.2 Mechanical Requirements

The following table lists the most important tests and limits for mechanical requirements:

| IEC/EN 61131-2 | Mechanical tests |
|---|---|
| | Vibration immunity test:<br>5...8,4 Hz, 3,5 mm<br>8.4...150 Hz, 1 g, EUT in operation, 10 cycles per axis |
| | Shock immunity test:<br>15 g, 11 ms, EUT in operation, 3 shocks per axis and direction (18 shocks) |

Table 6: Mechanical Tests

### 2.2.1.3 EMC Requirements

Higher interference levels are required for safety-related systems. HIMatrix systems meet these requirements in accordance with IEC 61131-6, IEC 61326-3-1, and IEC 61326-3-2. See column *Criterion FS* (Functional Safety).

| IEC/EN 61131-2 | Interference immunity tests | Criterion FS |
|---|---|---|
| IEC/EN 61000-4-2 | ESD test: 6 kV contact, 8 kV air discharge | 6 kV, 8 kV |
| IEC/EN 61000-4-3 | RFI test (10 V/m): 80 MHz...2 GHz, 80 % AM<br>RFI test (10 V/m): 2 GHz...3 GHz, 80 % AM<br>RFI test (20 V/m): 80 MHz...1 GHz, 80 % AM | 10 V/m<br>3 V/m<br>20 V/m |
| IEC/EN 61000-4-3 | 900 MHz pulses | - |
| IEC/EN 61000-4-4 | Burst test<br>Supply voltage: 2 kV<br>Signal lines: 2 kV | <br>4 kV<br>2 kV |
| IEC/EN 61000-4-5 | Surge:<br>Power lines: 2 kV CM, 1 kV DM<br>Signal lines: 2 kV CM, 1 kV DM at AC I/O | <br>2 kV /1 kV<br>2 kV |
| IEC/EN 61000-4-6 | High frequency, asymmetrical<br>10 V, 150 kHz...80 MHz, AM<br>20 V, ISM frequencies, 80 % AM | <br>10 V<br>- |
| IEC/EN 61000-4-12 | Damped oscillatory wave test<br>2.5 kV L-, L+ / PE<br>1 kV L+ / L- | <br>-<br>- |

Table 7: Interference Immunity Tests

| IEC/EN 61000-6-4 | Noise emission tests |
|---|---|
| EN 55011<br>Class A | Emission test:<br>radiated, conducted |

Table 8: Noise Emission Tests

#### 2.2.1.4 Power Supply

The following table lists the most important tests and limits for the HIMatrix systems' power supply:

| IEC/EN 61131-2 | Verification of the DC supply characteristics |
|---|---|
| | The power supply must comply with the following standards:<br>IEC/EN 61131-2:<br>SELV (Safety Extra Low Voltage) or<br>PELV (Protective Extra Low Voltage) |
| | HIMatrix systems must be fuse protected as specified in this manual |
| | Voltage range test:<br>24 VDC, -20...+25 % (19.2...30.0 V) |
| | Momentary external current interruption immunity test:<br>DC, PS 2: 2 ms |
| | Reversal of DC power supply polarity test:<br>Refer to corresponding chapter of the system manual or data sheet of power supply. |

Table 9: Verification of the DC Supply Characteristics

## 2.3 Tasks and Responsibilities of Operators and Machine and System Manufacturers

Operators and machine and system manufacturers are responsible for ensuring that HIMatrix systems are safely operated in automated systems and plants.

Machine and system manufacturers must sufficiently validate that the HIMatrix systems were properly programmed.

### 2.3.1 Connection of Communication Partners

Only devices with safe electrical separation may be connected to the communications interfaces.

### 2.3.2 Use of Safety-Related Communication

When implementing safety-related communications between various devices, ensure that the overall response time does not exceed the fault tolerance time. All calculations must be performed in accordance with the rules given in this chapter.

## 2.4 ESD Protective Measures

Only personnel with knowledge of ESD protective measures may modify or extend the system or replace a module.

**NOTE**

**Electrostatic discharge can damage the electronic components within the HIMatrix systems!**

- **When performing the work, make sure that the workspace is free of static, and wear an ESD wrist strap.**
- **If not used, ensure that the module is protected from electrostatic discharge, e.g., by storing it in its packaging.**

### 2.4.1 Noxious Gases

HIMatrix components may be operated without functional and safety restrictions in environments with noxious gas concentrations as described in the following standards:

- ANSI/ISA -S71.04:1985
  Corrosive gases, Class G3
- DIN EN 60068-2-60: 1996 (also IEC 68-2-60: 1995)

With noxious gas concentrations higher than those mentioned in the standards, a reduced component lifetime is to be expected. The user is responsible for demonstrating that the environment is sufficiently free from noxious gases.

## 2.5 Residual Risk

No imminent risk results from a modular HIMatrix M45 system itself.

Residual risk may result from:

- Faults related to engineering
- Faults related to the user program
- Faults related to the wiring

## 2.6 Safety Precautions

Observe all local safety requirements and use the protective equipment required on site.

## 2.7 Emergency Information

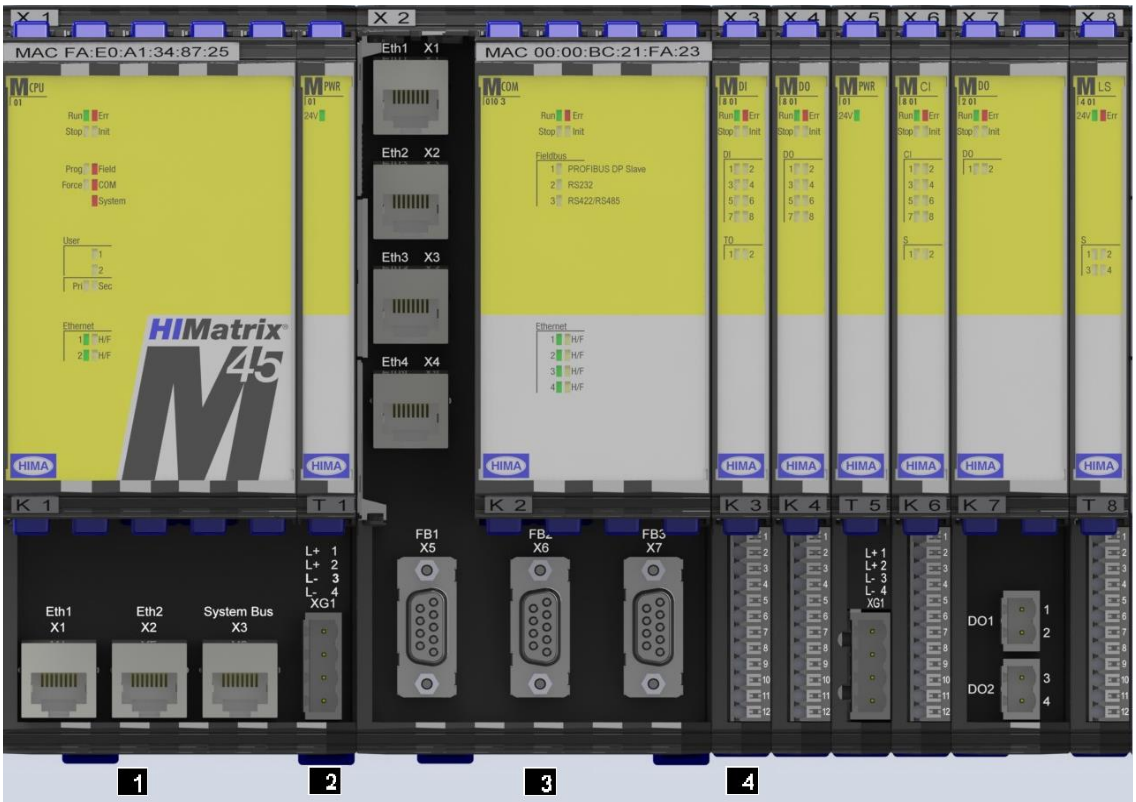
A HIMatrix M45 system is a part of the safety equipment of a plant. If a device or a module fails, the system enters the safe state.

In case of emergency, no action that may prevent the HIMatrix M45 systems from operating safely is permitted.

# 3 Product Description

The modular system HIMatrix M45 is composed of modules lined up on a DIN rail.

The only processor module is located at the left-hand side of the system. If required, one to three communication modules can be added to the right of the processor module. All additional modules are positioned at the right-hand side.




1 Processor Module M-CPU 01
2 Power Module M-PWR 01
3 Communication Module M-COM 01
4 I/O Modules from Here to the Right

Figure 1: HIMatrix M45 System

The modules are inserted into sockets. The sockets are fastened to the DIN rail. This allows one to replace modules without removing the field wiring.

The sockets have a lateral electrical connection for attaching modules to the system bus and supply voltage.

The socket of the processor module also accommodates the M-PWR 01 power module 2, which is the minimum requirement for supplying the controller with electrical power.

The M-PWR 01 power module in the processor module socket is sufficient for small controllers. If more power or isolated operation of I/O modules is required, additional power modules can be used on the DIN rail to supply the I/O modules located to the right.

The modular M45 system can also be connected to HIMatrix F60 and compact systems via safe**ethernet**.

## 3.1 Modules

All HIMatrix M45 system modules are 165 mm in height.

Certain slots are specified for the first power module, the processor module, and communication modules if used, see above.

**NOTE**

**Controller damage!**

**Inserting and removing the modules during operation is not permitted.**

**Prior to modifying the assembly, ensure that the controller is shut down!**

After modifying a controller assembly, its user program must be adapted to the change and reloaded.

Pluggable clamps located on the modules' sockets are used to connect sensors and actuators. The modules indicate the status of digital signals via LEDs located on the front plate.

### 3.1.1 Inputs

The modules' input channels are used to transmit and adjust the signals between the sensors and the microprocessor system on the CPU module.

The safety-related modules continuously test themselves. If a fault occurs, the module restarts. If the same fault occurs again within one minute, the module shuts down and displays the fault. The controller's operating system then sets the user program's input value to the initial value.

Instead of contacts, signal sources with own voltage may also be connected. To this end, the signal source ground must be connected to the input ground.

#### 3.1.1.1 Surges on Digital Inputs

Due to the short cycle time of the HIMatrix M45 systems, a surge pulse as described in EN 61000-4-5 can be read in to the digital inputs as a short-term high level.

To prevent malfunctions, take one of the following measures for the application:

- Install shielded input wires to prevent surges within the system.
- Activate noise blanking: a signal must be present for at least two cycles before it is evaluated.

  Caution: This action increases the system's response time!

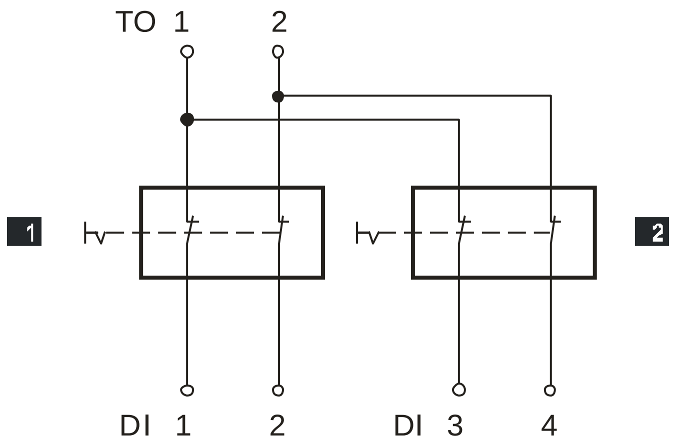
i The measures specified above are not necessary if the plant design precludes surges from occurring within the system.

In particular, the design must include protective measures with respect to overvoltage, lightning, earth grounding and plant wiring in accordance with the relevant standards and the manufacturer's specifications.

#### 3.1.1.2 Line Control

Line control is used to detect short-circuits or open-circuits, e.g., on EMERGENCY STOP inputs complying with Cat. 4 and PL e in accordance with EN ISO 13849-1. Line control can be configured for the M45 system.

Application example: The outputs TO 1 through TO 2 of the M-DI 8 01 module are connected to the digital inputs (DI) of the same module as follows:




**1** EMERGENCY STOP 1
**2** EMERGENCY STOP 2

EMERGENCY STOP devices in accordance with EN 60947-5-1 and EN 60947-5-5

Figure 2: Line Control

The digital outputs are pulsed. This allows monitoring of the wires to the digital inputs of the M-DI 8 01 module.

A fault reaction is triggered if one of the following faults occurs:

- Cross-circuit between two parallel wires.
- Improper connections of two lines (e.g., TO 2 to DI 3).
- Earth fault of a line (with earthed ground only).
- Open-circuit or open contacts, i.e., including when one of the two EMERGENCY STOP switches mentioned above has been engaged.

The fault reaction includes the following actions:

- The *ERROR* LED on the controller's front plate is blinking.
- The inputs are set to 0.
- An evaluable error code is created.

For more information on how to configure line control in the user program, refer to the module-specific HIMatrix M45 manuals.

## 3.1.2 Outputs

The module's output channels are used to transmit and adjust the signals between the microprocessor systems on the central module and the actuators.

The safety-related modules continuously test themselves. If a fault occurs, the module restarts. If the same fault occurs again within one minute, the module shuts down and displays the fault. The outputs are set to the safe de-energized state.

### NOTE

**Controller damage!**

**Plug the terminals for output circuits without a load.**

- **If short-circuits are present, the resulting high current may damage the terminals.**

Inductive loads may be connected with no free-wheeling diode on the actuator. However, HIMA recommends connecting a protective diode directly to the actuator.

For more details on line control, refer to Chapter 3.1.1

## 3.2 Extending the System Bus

Systems composed of a larger or the maximum number of modules require a very long DIN rail. This can result in exceeding the width of the planned mounting space. In this case, multiple short DIN rails may replace a long one. Short rails are usually mounted one above another. An Ethernet cable connects the system bus between two rails. The Ethernet cable is connected to the system bus on either side via an extension socket. The cable connects the right extension socket, M-SO EXT 02, of the upper rail to the left extension socket, M-SO EXT 01, of the lower rail.

Only use passive Ethernet cables to extend the system bus.

The maximum length of the Ethernet cable between two extension sockets is 100 m. An M45 system bus may be extended via Ethernet cable at no more than 10 places. The maximum overall length of the Ethernet cables may be 1000 m.

Lay the Ethernet cables ensuring their adequate protection. They must not be run parallel to power supply cables.

A power module must be used after the left extension socket on each rail.

## 3.3 Supply Voltage Monitoring

The HIMatrix system is a single voltage system. In accordance with IEC/EN 61131-2, the required supply voltage is defined as follows:

| Supply voltage | |
|---|---|
| Nominal value | 24 VDC, -15...+20 %<br>20.4...28.8 V |
| Max. permissible function limits in continuous operation | 18.5...30.2 V<br>(including ripple) |
| Permissible ripple | r < 5 % as r.m.s. value<br>$r_{PP}$ < 15 % as value peak-to-peak |
| Ground | L- (negative pole)<br>Earthing the ground is permitted, see Chapter Earthing. |

Table 10: Supply Voltage

The power supply units of HIMatrix systems must comply with the SELV (Safety Extra Low Voltage) or PELV (Protective Extra Low Voltage) requirements. At the power supply units' outputs, a maximum overvoltage of 35 V may occur in case of fault.

Observing the permitted voltage limits guarantees the system's proper operation.

The required SELV/PELV power supply units ensure safe operation.

The device monitors the 24 VDC voltage during operation. Reactions occur in accordance with the specified voltage level:

| Voltage level | Reaction of the device |
|---|---|
| 19.3...28.8 V | Normal operation |
| < 18.0 V | Alarm states (internal variables are written and provided to the inputs or outputs) |
| < 12.0 V | Switching off the inputs and outputs |

Table 11: Operating Voltage Monitoring

The *Power Supply State* system variable is used to evaluate the operating voltage state with the programming tool or from within the user program.

## 3.4 Monitoring the Temperature State

One or multiple sensors are used to measure the temperature at relevant positions within the module.

If the measured temperature exceeds the defined temperature threshold, the value of the *Temperature State* system variable changes as follows:

| Temperature | Temperature range | *Temperature State* [BYTE] |
|---|---|---|
| < 60 °C | Normal | 0x00 |
| 60 °C…70 °C | High temperature | 0x01 |
| > 70 °C | Very high temperature | 0x03 |
| Back to 64 °C...54 °C [1)] | High temperature | 0x01 |
| Back to < 54 °C [1)] | Normal | 0x00 |
| [1)] The hysteresis of sensors is 6 °C. | | |

Table 12: Temperature Monitoring

If no or insufficient air circulates within a control cabinet and natural convection is not enough, the threshold associated with *High Temperature* in the HIMatrix M45 controller can already be exceeded at ambient temperatures of less than 35 °C.

This can be due to local heating or to a bad heat conduction. In particular with digital outputs, the heat levels strongly depend on their load.

The *Temperature State* system variable allows the user program to read the temperature. If the state *Very high temperature* often occurs, HIMA recommends improving the system heat dissipation, e.g., by taking additional ventilation or cooling measures, such that the long life time of the HIMatrix systems can be maintained.

i The safety of the system is not compromised if the state *High Temperature* or *Very High Temperature* is entered.

### 3.4.1 Setting the Temperature Threshold for Messages

The temperature threshold that, if exceeded, causes a message to be issued can be defined for the M45 system. In the SILworX Hardware Editor, use the detail view for the rack to configure this setting.

## 3.5 Short-Circuit Reaction of Output Channels

If a short-circuit occurs in an output channel, the HIMatrix automation systems switch off the affected channel. If multiple short-circuits occur, the channels are switched off individually in accordance with their current input.

If the maximum current permitted for all outputs is exceeded, all outputs are switched off and cyclically switched on again.

i The terminals for output circuits must be plugged without a load. If short-circuits are present, the resulting high current may damage the terminals.

## 3.6 Alarms&Events Recording

The HIMatrix M45 system is able to record alarms and sequences of events (SOE).

## 3.6.1 Alarms&Events

Events are state changes of a variable that are performed by the plant or controller, and are provided with a timestamp.

Alarms are events that signalize increased risk potential.

The HIMatrix system records the state changes as events specifying the time point when they occurred. The X-OPC server transfers the events to other systems such as control systems that display or evaluate the events.

HIMatrix differentiates between Boolean and scalar events.

Boolean events:

- Changes of Boolean variables, e.g., of digital inputs.
- Alarm and normal state: They can be arbitrarily assigned to the variable states.

Scalar events:

- Crossing the upper and lower limit values defined for a scalar variable.
- Scalar variables have a numeric data type, e.g., INT, REAL.
- Two upper limits and two lower limits are possible.
- The following condition must be met for the limits:
  Highest limit (HH) ≥ high limit (H) ≥ normal range ≥ low limit (L) ≥ lowest limit (LL).
- A hysteresis can be effective in the following cases:
  - If the value falls below one of the upper limits.
  - If the value exceeds one of the lower limit.

A hysteresis is defined to avoid a needless large number of events when a global variable strongly oscillate around a limit.

HIMatrix M45 can only create events if they are configured in SILworX, see Chapter 7.8. Up to 4 000 alarms and events can be defined.

## 3.6.2 Creating Events

The processor system is able to create events.

The processor system uses global variables to create the events and stores them in the buffer, see Chapter 3.6.3. The events are created in the user program cycle.

Every event that has been read can be overwritten by a new event.

### System Events

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

- Overflow: Some events were not stored due to buffer overflow. The timestamp of the overflow event corresponds to that of the event causing the overflow.
- Init: The event buffer was initialized.

System events contain the SRS identifier of the device causing the events.

### Status Variables

Status variables provide the user program with the state of scalar events. Each of the following states is connected to a status variable and can be assigned a global variable of type BOOL:

- Normal.
- Low limit (L) exceeded.
- Lowest limit (LL) exceeded.
- High limit (H) exceeded.
- Highest limit (HH) exceeded.

The assigned status variable becomes TRUE when the corresponding state is achieved.

### 3.6.3 Recording Events

The processor system records the events:

The processor system stores all the events in its buffer. The buffer is part of the non-volatile memory and has a capacity of 1 000 events.

If the event buffer is full, no new events can be stored as long as no further events are read and thus marked as to be overwritten.

### 3.6.4 Transfer of Events

The X-OPC server reads the events from buffer and transfers them to a third-party system for evaluation and representation purposes. Four X-OPC servers can simultaneously read events out of a processor module.

## 3.7 Product Data

| Designation | Value, range of values |
|---|---|
| Power supply<br>Power supply module | 24 VDC, -15 %...+20 %, $r_{PP} \leq 15$ %,<br>externally fused with 32 A |
| | Goldcap in the processor module<br>(for buffering date/time) |
| Maximum supply voltage | 30 V |
| Operating temperature | 0...+60 °C |
| Storage temperature | -40...+85 °C |
| Type of protection | IP20 |
| Number of processor modules | 1 |
| Number of I/O modules | 1...62, including the communication modules |
| Number of communication modules | 0…3 |
| Number of power modules | 1…62 |

Table 13: Specifications HIMatrix M45

The specifications for the modules are detailed in the corresponding manuals.

# 4 Communication

Communication runs via the following interfaces:

- Ethernet interfaces
- Fieldbus interfaces

## 4.1 HIMatrix Communication Protocols M45

Depending on the HIMatrix controller and its interfaces, different communication protocols can be activated:

1. safe**ethernet** and SNTP are activated by default in all HIMatrix systems.
2. Communication via serial interfaces requires an additional license (software activation code). For further information, refer to the SILworX communication manual (HI 801 101 E).
3. All Ethernet protocols can be tested without software activation code for 5000 operating hours.

i After 5000 operating hours, communication continues until the controller is stopped. Afterwards, the user program cannot be started without a valid license for the protocols used in the project (invalid configuration).

Order the software activation code on time!

The software activation code can be generated on the HIMA website using the system ID of the controller (value 1...65 535).

The software activation code is intrinsically bound to this system ID. One license can only be used one time for a specific system ID. For this reason, only activate the code when the system ID has been uniquely defined.

HIMatrix M45 systems support the following Ethernet interface communication protocols.

- safe**ethernet**, redundant operation possible
- Modbus TCP master
- Modbus TCP slave
- SNTP

Each protocol can be used once per controller.

Communication options for serial interfaces are described in Chapter 4.3.

## 4.2 Ethernet Communication

### 4.2.1 Communication via Switches

The HIMatrix M45 controllers are equipped with Ethernet switches with RJ-45 connectors to which Ethernet cables can be attached for connecting to other devices:

i **When configuring safety-related communication, observe the instructions specified in the safety manual (HI 800 653 E).**

### 4.2.2 safe**ethernet**

An overview of safe**ethernet** is available in the corresponding chapter of the SILworX communication manual (HI 801 101 E).

The different systems can be connected to one another via Ethernet in any configuration (e.g., star or linear network); a PADT may also be connected to any device.

**NOTE**

**Ethernet operation may be disturbed!**

**Ensure that no network rings result from interconnecting the controllers. Data packets may only reach a system over a single path.**

If M45 controllers are connected to remote I/Os via safe**ethernet**, ensure that the remote I/O is equipped with a CPU operating system V7 or higher.

### 4.2.3 Maximum Communication Time Slice

The maximum communication time slice is the time period in milliseconds (ms) and per cycle assigned to the processor system for processing the communication tasks.
If not all upcoming communication tasks can be processed within one cycle, the whole communication data is transferred over multiple cycles (number of communication time slices > 1).

i When calculating the maximum response times allowed, the number of communication time slices must be equal to 1, see the communication manual (HI 801 101 E). The duration of the communication time slice must be set such that, when using the communication time slice, the cycle cannot exceed the watchdog time specified by the process.

### 4.2.4 Connectors for safe**ethernet**/Ethernet

The connectors for Ethernet communication are located in the socket of the processor module and of the communication module.

- The processor module contains three RJ-45 connectors labeled *Ethernet1*, *Ethernet2*, and *Local*.
- The communication module has four RJ-45 connectors *Eth1*…*Eth4*.

To connect the HIMatrix M45 systems, only interference-free Ethernet cables may be used, e.g., shielded (STP)!

### 4.2.5 Communication with the PADT

A HIMatrix controller communicates with a PADT via Ethernet. A PADT is a computer that is installed with the SILworX programming tool.

The computer must be able to reach the controller via Ethernet.

A controller can simultaneously communicate with up to 5 PADTs. If this is the case, only one programming tool can access the controller with write permission. The remaining programming tools can only read information. If they try to establish a writing connection, the controller only allows them a read-only access.

### 4.2.6 Ethernet Communication Protocols

In addition to the safe**ethernet**, HIMatrix supports the following Ethernet communication protocols:

- SNTP
- Modbus TCP
- Send & Receive TCP
- PROFINET IO and PROFIsafe

For details of the different protocols, see the communication manual (HI 801 101 E).

#### 4.2.6.1 SNTP

The SNTP protocol (simple network time protocol) is used to synchronize the time of the HIMA resources via Ethernet. The current time can be retrieved via Ethernet in predefined time intervals from a PC, or a HIMA resource configured as SNTP server.

HIMA resources with COM OS V6 or higher, can be configured and used as SNTP server and/or as SNTP client. The SNTP server communicates with the SNTP client via the non-safe UDP protocol on port 123.

For further details on the SNTP protocol, refer to the SILworX communication manual (HI 801 101 E) or the online help of the programming tool.

#### 4.2.6.2 Modbus TCP

The HIMA-specific designation for the **non**-safety-related Modbus TCP is Modbus Master/Slave Eth.

The fieldbus protocols Modbus master/slave can communicate with the Modbus TCP via the Ethernet interfaces of the HIMatrix controllers.

In a standard Modbus communication, the slave address and a CRC checksum are transferred in addition to the instruction code and data, while in a Modbus TCP, this function is assumed by the subordinate TCP protocol.

For further details on the Modbus TCP, refer to the SILworX communication manual (HI 801 101 E) or HIMatrix Modbus master/slave manual (HI 800 003 E).

#### 4.2.6.3 Send & Receive TCP

S&R TCP is a manufacturer-independent, **non**-safety-related protocol for cyclic and acyclic data exchange and does not use any specific protocols other than TCP/IP.

With S&R TCP, HIMatrix systems are able to support almost every third-party system as well as PCs with implemented socket interface to TCP/IP (e.g., Winsock.dll).

For further information on the S&R protocol, refer to the SILworX communication manual (HI 801 101 E).

#### 4.2.6.4 PROFINET IO and PROFIsafe

The non-safety-related protocol PROFINET IO and the safety-related protocol PROFIsafe must be configured using SILworX. Refer to the SILworX communication manual (HI 801 101 E) for more information on communication.

## 4.3 Fieldbus Communication

The communication module is equipped with three fieldbus communication terminals. These 9-pole D-sub connectors are located on the front plate of the module.

The two interfaces can operate simultaneously.

**NOTE**

**Fieldbus operation of other devices may be disturbed!**

**If the controller is reset, do not connect the fieldbus interfaces to an operational fieldbus to prevent potential disturbances.**

### 4.3.1 Protocols of the Fieldbus Interfaces

The protocols supporting the fieldbus interfaces FB1...FB3 of the communication module depend on the type of the communication module.

Table 14 shows the types of communication protocols and the fieldbus interfaces supported by them. Refer to the communication module-specific manual (HI 800 657 E) for further details.

| Type | Protocols on the interfaces | | |
|---|---|---|---|
| | FB1 | FB2 | FB3 |
| M-COM 010 2 | PROFIBUS DP master | RS485 | RS422/RS485 |
| M-COM 010 3 | PROFIBUS DP slave | RS232 | RS422/RS485 |
| M-COM 010 7 | CAN[1)] | RS485 | RS422/RS485 |
| M-COM 010 8 | SSI | RS485 | RS422/RS485 |
| [1)] Use HIMA CAN interface cable | | | |

Table 14: Fieldbus Submodule

The fieldbus protocols are selected when ordering the controller using the part number.

Depending on the type of communication module, the communication protocols must be activated. For further information on how to register and activate the protocols, refer to the SILworX communication manual (HI 801 101 E).

### 4.3.2 Restrictions for Simultaneous Operation of Protocols

- PROFIBUS DP master or slave can only be operated on one fieldbus interface, i.e., two PROFIBUS masters or slaves may not be operated at a time within a resource and, therefore, they will not function.
- Modbus master/slave RS485 can only be operated on one fieldbus interface. However, simultaneous operation via RS485 and Ethernet is allowed.

i No safety-related communication over fieldbus interfaces can be ensured with the currently available protocols.

Only devices with safe electrical separation may be connected to the fieldbus interfaces.

# 5 Operating System

The operating system includes all basic functions of the HIMatrix M45 controller ( PES).

Which user functions the PES should perform is specified in the user program. A code generator translates the user program into a machine code. The programming tool transfers this machine code to the controller's flash memory.

## 5.1 Functions of the Processor Operating System

The following table specifies all basic functions of the operating system for a processor system and the connections to the user program:

| Functions of the operating system | Connections to the user program |
|---|---|
| Cyclic processing the user program | It affects variables, function blocks |
| Automation device configuration | Defined by selecting the controller |
| Processor tests | - - - |
| I/O module tests | Depending on the type |
| Reactions in the event of a fault: | Preset and not changeable<br>The user program is responsible for the process reaction |
| Processor system and I/O diagnosis | Use of system signals/variables for error messages |
| Safe communication:<br>peer-to-peer<br>Non-safe communication:<br>PROFIBUS DP, Modbus | To define the use of communication signals/variables |
| PADT interface:<br>Actions allowed | Defined in the programming tool:<br>Configuration of protective functions,<br>User log-in |

Table 15: Functions of the Processor Operating System

Each operating system is inspected by the TÜV in charge, and approved for operation in the safety-related controller. The valid versions of the operating system and corresponding signatures (CRCs) are documented in a list maintained by HIMA in co-operation with the TÜV.

Additional features of one operating system version compared to the previous version can only be exploited if a suitable version of the programming tool is used.

## 5.2 Indication of the Operating System Versions

The current COM and CPU operating system versions can be displayed using the module data overview, see the SILworX online help. The module data overview is activated from within the online view of the Hardware Editor, selecting the menu option **Online**.

The *OS* column contains the list of the current operating system versions.

## 5.3 Behavior in the Event of Faults

The reaction to faults  detected during tests is important. The distinction between the following types of faults is made:

- Permanent faults on inputs or outputs
- Temporary faults on inputs or outputs
- Internal faults

### 5.3.1 Permanent Faults  on Inputs or Outputs

A fault on an input or output channel has not effect on the controller. The operating system only considers the defective channel as faulty, and not the entire controller. The remaining safety functions are not affected and remain active.

If the input channels are faulty, the operating system sends the safe value 0 or the initial value for further processing.

Faulty output channels are set to the de-energized state by the operating system. If it is not possible to just switch off a single channel, the entire output module is considered as faulty.

The operating system sets the fault status signal and reports the type of fault to the user program.

If the controller is not able to switch off a given output and even the second switch-off option is not effective, the controller enters the STOP state. The outputs are then switched off by the watchdog of the processor system.

If faults are present in the I/O modules for longer than 24 hours, only the affected I/O modules are permanently switched off by the controller.

### 5.3.2 Temporary Faults on Inputs or Outputs

If a fault occurs in an input or output module and disappears by itself, the operating system resets the fault status and resumes normal operation.

The operating system statistically evaluates the frequency with which a fault occurs. If the specified fault frequency is exceeded, it permanently sets the module status to *faulty*. In this way, the module no longer operates, even if the fault disappears. The module is released and the fault statistics are reset when the controller operating state switches from STOP to RUN. This change acknowledges the module fault.

### 5.3.3 Internal Faults

In the seldom case that a HIMatrix M45 controller detects an internal fault, it restarts automatically. Should an internal fault be detected again within the first minute after start up, the HIMatrix M45 controller will remain in the STOP/INVALID CONFIGURATION state.

## 5.4 Processor Module

The processor module is the central component of the controller and communicates with the I/O modules of the controller via the system bus.

The processor module monitors the sequence and the proper logical execution of the operating system and user program. The following functions are monitored with respect to time:

- Processor module hardware and software self-tests
- Processor module RUN cycle (including the user program)
- I/O tests and processing of I/O signals

### 5.4.1 Operating States of the Processor Module

LEDs located on the front plate of the controller indicate the operating state of the processor module. The latter can also be reported by the PADT, together with other parameters specific to processor module and user program.

Stopping the processor interrupts the execution of the user program and sets the outputs of the controller and all remote I/Os to safe values.

If the user program sets the EMERGENCY STOP system variables to TRUE, the processor module enters the STOP state.

The following table specifies the most important operating states:

| Mode of operation | Description |
|---|---|
| INIT | Safe state of the processor module during the initialization phase.<br>Hardware and software tests are performed. |
| STOP / VALID CONFIGURATION | Safe state of the processor module with no execution of the user program<br>All outputs of the controller are reset.<br>Hardware and software tests are performed. |
| STOP / INVALID CONFIGURATION | Safe state of the processor module without a configuration loaded or after a system fault.<br>All controller's outputs are reset, the hardware watchdog has not triggered.<br>The processor module can only be rebooted using the PADT. |
| RUN | The processor module is active:<br>The user program is run,<br>I/O signals are processed.<br>The processor module performs safety-related and non-safety-related communication (if configured).<br>Hardware and software tests, and test for configured I/O modules are performed. |

Table 16: Modes of Operation for the Processor Module

### 5.4.2 Programming

A PADT (programming and debugging tool) is used to program the HIMatrix M45 controllers. The PADT is a PC equipped with SILworX.

SILworX supports the following programming languages in accordance with IEC 61131-3:

- Function block diagrams (FBD)
- Sequential function charts (SFC)
- Structured text (function blocks)

SILworX is suitable for developing safety-related programs and for operating the controllers.

Refer to the SILworX online help and SILworX first steps manual (HI 801 103 E) for further details.

# 6 User Program

In accordance with the IEC 61131-3 requirements, a PADT with installed programming tool SILworX must be used to create and load the user program for the PES.

First, use SILworX to create and configure the user program for the controller's safety-related operation. To this end, observe the instructions specified in the safety manual (HI 800 653 E) and ensure that the requirements specified in the report to the certificate are met.

Upon completion of compiling, SILworX loads the user program (logic) and its configuration (connection parameters such as IP address, subnet mask and system ID) into the controller and starts them.

SILworX can be used to perform the following actions while the controller is operating:

- Starting and stopping the user program.
- Displaying and forcing variables or signals using the Force Editor.
- In test mode, executing the user program in single steps - not suitable for safety-related operation.
- Reading the diagnostic history.

This is possible, provided that SILworX and the controller are loaded with the same user program.

## 6.1 Modes of Operation for the User Program

Only one user program at a time can be loaded into a given controller. For this user program, the following modes of operations are allowed:

| Mode of operation | Description |
|---|---|
| RUN | The processor module is in RUN.<br>The user program is run cyclically.<br>I/O signals are processed. |
| Test Mode<br>(single step) | The processor module is in RUN.<br>The user is run cyclically, if previously set by the user.<br>I/O signals are processed.<br>**Do not use this function during safety-related operation!** |
| STOP | The processor module is in STOP.<br>The user program is not or no longer run, the outputs are reset. |
| ERROR | A loaded user program has been stopped due to a failure.<br>The outputs are reset.<br>Note: The program can only be restarted using the PADT. |

Table 17: User Program Modes of Operation

## 6.2 Cycle Sequence

In a simplified overview, the processor module cycle (CPU cycle) of only one user program runs through the following phases:

1. Processing of the input data
2. Processing of the user program
3. Provision of the output data

These phases do not include special tasks such as reload, which might be executed within a CPU cycle.

Global variables, results from function blocks, and other data are processed in the first phase and represent the input data for the second phase. The first phase need not start at the beginning of the cycle, but may be delayed. For this reason, using timer function blocks to determine the cycle time in the user program may result in inaccurate cycle times, potentially exceeding the watchdog time.

In the third phase, the user program results are forwarded for being processed in the following cycles and supplied to the output channels.

### 6.2.1 Multitasking

Multitasking refers to the capability of the HIMatrix system to process up to 32 user programs within the processor module.

This allows the project's sub-functions to be separated from one another. The individual user programs can be started and stopped independently from one another. SILworX displays the states of the individual user programs in the Control Panel and enables their operation.

Using multitasking, the second phase changes so that a CPU cycle performs the following tasks:

1. Processing of the input data
2. Processing of all the user programs
3. Provision of the output data

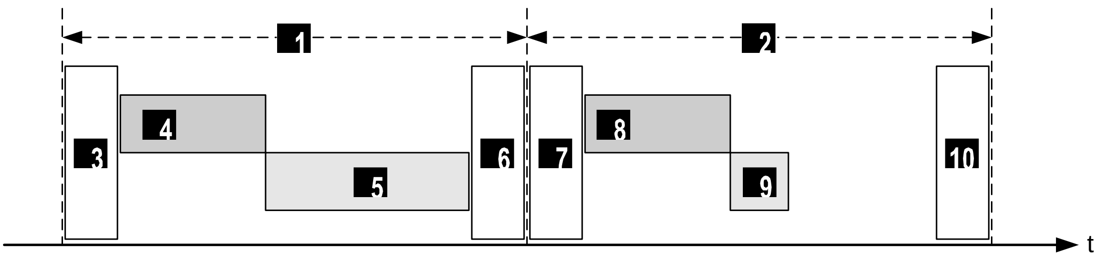
In the second phase, the HIMatrix can run up to 32 user programs. Two scenarios are possible for each user program:

- An entire user program cycle can be run within a single CPU cycle.
- A user program cycle requires multiple CPU cycles to be completed.

These two scenarios are even possible if only **one** user program exists.

It is not possible to exchange global data between user programs within a single CPU cycle. Data written by a user program is provided immediately before phase 3, but after the user program has been completely executed. This data can thus first be used as input values at the next start of another user program cycle.

The example in Figure 3 shows both scenarios in a project containing two user programs, Prg 1 and Prg 2.




1 First CPU Cycle Considered
2 Second CPU Cycle Considered
3 Input Processing in the First CPU Cycle
4 First Prg 1 Cycle Considered
5 First Portion of the Considered Prg 2 Cycle
6 Output Processing in the First CPU Cycle
7 Input Processing in the Second CPU Cycle
8 Second Prg 1 Cycle Considered
9 Second Portion of the Considered Prg 2 Cycle
10 Output Processing in the Second CPU Cycle

Figure 3: CPU Cycle Sequence with Multitasking

Each Prg 1 cycle is completely processed during each CPU cycle. *Prg 1* processes an input change registered by the system at the beginning of the CPU cycle 1 and delivers a reaction at the end of the cycle.

One Prg 2 cycle requires two CPU cycles to be processed. Prg 2 needs CPU cycle 1 to process an input change registered by the system at the beginning of CPU cycle 2. For this reason, the reaction to this input change is only available at the end of CPU cycle 2.
The reaction time of Prg 2 is two times longer than that of Prg 1.

Upon completion of the first part 5 of the Prg 2 cycle under consideration, Prg 2 processing is completely aborted and only resumed when 9 starts. During its cycle, Prg 2 processes the data provided by the system during 3. The results of Prg 2 are available to the system during 10 (e.g., for process output). The data that the system exchanges with the user program are always consistent.

The program execution order can be controlled by assigning a priority, which indicates how important the corresponding user program is compared to the others (see multitasking mode 2).

To specify the user program execution order, use the following parameters in the resources and programs or in the Multitasking Editor:

i A license is required to use the multitasking feature.

| Parameter | Description | Configurable for |
|---|---|---|
| Watchdog Time | Resource watchdog time | Resource, Multitasking Editor |
| Target Cycle Time [ms] | Required or maximum cycle time | Resource, Multitasking Editor |
| Multitasking Mode | Use of the execution duration unneeded by the user program, e. g., the difference between actual execution duration in one CPU cycle and the defined *Max. Duration for Each Cycle [µs]*. | Resource, Multitasking Editor |
| | Mode 1 The duration of a CPU cycle is based on the required execution time of all user programs. | |
| | Mode 2 The processor provides user programs with a higher priority the execution time not needed by user programs with a lower priority. Operation mode for high availability. | |
| | Mode 3 The processor waits until the execution time not needed by the user programs has expired, thus increasing the cycle. | |
| Target Cycle Time Mode | Use of *Target Cycle Time [ms].* | Resource, Multitasking Editor |
| Program ID | ID for identifying the program when displayed in SILworX | User program |
| Priority | Importance of a user program; highest priority: 0. | User program |
| Program's Maximum Number of CPU Cycles | Maximum number of CPU cycles required to process one user program cycle. | User program |
| Max. Duration for Each Cycle [µs] | Time permitted for executing the user program within a CPU cycle. | User program |

Table 18: Parameters Configurable for Multitasking

Observe the following rules when setting the parameters:

- If *Max. Duration for Each Cycle [µs]* is set to 0, the execution time of the user program is not limited, e.g., it is always processed completely. Therefore, the number of cycles may be set to 1 in this case.
- The sum of the *Max. Duration for Each Cycle [µs]* parameters in all user programs must not exceed the resource watchdog time. Make sure that sufficient reserve is planned for processing the remaining system tasks.

- The sum of the *Max. Duration for Each Cycle [µs]* parameters in all user programs must be large enough to ensure that sufficient reserve is available to maintain the target cycle time.
- The *Program ID*s of all user programs must be unique.

During verification and code generation, SILworX monitors that these rules are observed. These rules must also be observed when modifying the parameters online.

SILworX uses these parameters to calculate the user program watchdog time:
User program watchdog time = *watchdog time * maximum number of cycles*

The sequence control for executing the user programs is run in cycles of 250 µs. For this reason, the values set for *Max. Duration For Each Cycle [µs]* can be exceeded or under-run by up to 250 µs.

Usually, the individual user programs operate interference-free and independently to one another. However, reciprocal influence can be caused by:

- Use of the same global variables in several user programs.
- Unpredictably long runtimes can occur in individual user programs if a limit is not configured with *Max. Duration for Each Cycle [µs]*.

**NOTE**

**Reciprocal influence of user programs is possible!**

**The use of the same global variables in several user programs can lead to a variety of consequences caused by the reciprocal influence among the user programs.**

- **Carefully plan the use of the same global variables in several user programs.**
- **Use the cross-references in SILworX to check the use of global data. Global data may only be assigned values in one location, either in a user program or from the hardware!**

HIMA recommends setting the *Max. Duration for each Cycle [µs]* parameter to an appropriate value ≠ 0. This ensures that a user program with an excessively long runtime is stopped during the current CPU cycle and resumed in the next CPU cycle without affecting the other user programs.

Otherwise, an unusually long runtime for one or several user programs can cause the target cycle time, or even the resource watchdog time, to be exceeded, thus leading to an error stop of the controller.

The operating system defines in which order the user programs are executed in accordance with the following scheme:

- User programs with lower priority are executed before user programs with higher priority.
- If the user programs have the same priority, the system processes them in ascending order of the *Program ID*s.

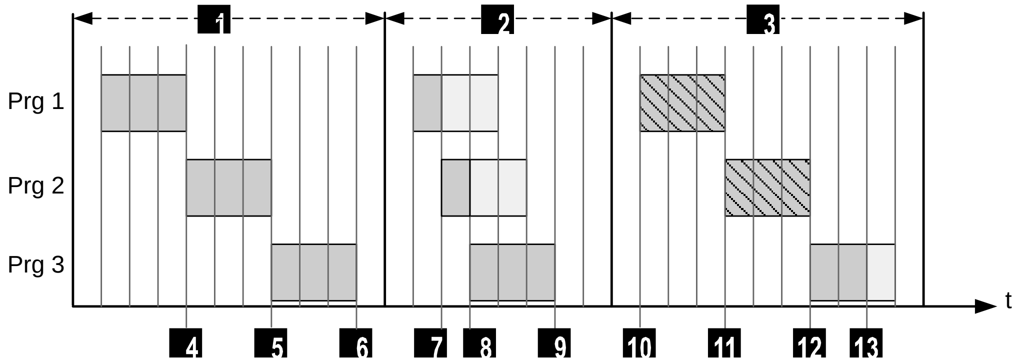
This order is also followed when starting and stopping the user program during the start and stop of the PES, respectively.

### 6.2.2 Multitasking Mode

Three operation modes exist for multitasking. These modes differ in how the time that is not needed for executing the CPU cycle of the user programs is used. One of these three modes can be selected for every resource.

1. **Multitasking Mode 1** uses the unneeded time to reduce the CPU cycle. If the user program is completely processed, processing of the next user program begins immediately. In total, this results in a shorter cycle.

Example: 3 user programs (*Prg 1*, *Prg 2* and *Prg 3*) that allow a user program cycle to take up to 3 CPU cycles.



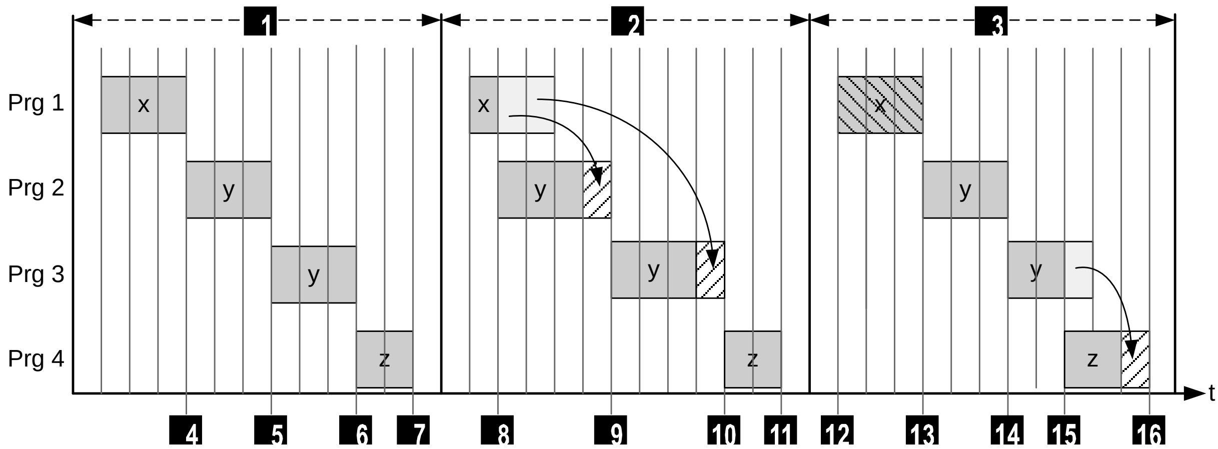

- **1** First CPU Cycle Considered.
- **2** Second CPU Cycle Considered.
- **3** Third CPU Cycle Considered.
- **4** *Max. Duration for Each Cycle [µs]* of *Prg 1* has Expired, *Prg 2* Starts.
- **5** *Max. Duration for Each Cycle [µs]* of *Prg 2* has Expired, *Prg 3* Starts.
- **6** *Max. Duration per Cycle [µs]* of *Prg 3* has Expired, End of the First CPU Cycle.
- **7** Completion of the *Prg 1* cycle, *Prg 2* is Resumed.
- **8** Completion of the *Prg 2* Cycle, *Prg 3* is Resumed.
- **9** *Max. Duration per Cycle [µs]* of *Prg 3* has Expired, End of the Second CPU Cycle.
- **10** The Next User Program Cycle *Prg 1* Starts.
- **11** *Max. Duration for Each Cycle [µs]* of *UP 1* has Expired. Next User Program Cycle of *Prg 2* Starts.
- **12** *Max. Duration for Each Cycle [µs]* of *Prg 2* has Expired, *Prg 3* Starts.
- **13** Completion of the User Program Cycle *Prg 3*.

Figure 4: Multitasking Mode 1

2. In **Multitasking Mode 2,** the unneeded duration of lower-priority user programs is distributed among higher-priority user programs. In addition to the specified *Max. Duration for Each Cycle [µs],* these user programs can use the portions of unneeded duration. This procedure ensures high availability.

   Four user programs are used in the example: *Prg 1*…*Prg 4*. The following priorities are allocated to the user programs:

   - *Prg 1* has the lowest priority, priority *x*
   - *Prg 2* and *Prg  3* have a medium priority, priority *y*
   - *Prg 4* has the highest priority, priority *z*




**1** First CPU Cycle Considered.

**2** Second CPU Cycle Considered.

**3** Third CPU Cycle Considered.

**4** *Max. Duration for Each Cycle [µs]* of *Prg 1* has Expired, Prg 2 Starts.

**5** *Max. Duration for Each Cycle [µs]* of *Prg 2* has Expired, *Prg 3* Starts.

**6** *Max. Duration for Each Cycle [µs]* of *Prg 3* has Expired, *Prg 4* Starts.

**7** *Max. Duration for Each Cycle [µs]* of *Prg 4* has Expired, Completion of the First CPU Cycle.

**8** Completion of the User Program Cycle of *Prg 1*, *Prg 2* is Resumed. The Remaining Duration is Distributed to the *Max. Duration for Each Cycle [µs]* of *Prg 2* and *Prg 3* (Medium Priority *y*) (Arrows).

**9** *Max. Duration for Each Cycle [µs]* of *Prg 2* + Proportional Remaining Duration of *Prg 1* Expired, *Prg 3* is Resumed.

**10** *Max. Duration per Cycle [µs]* of *Prg 3* and Proportional Remaining Duration of *Prg 1* has Expired, *Prg 4* Starts.

**11** *Max. Duration for Each Cycle [µs]* of *Prg 4* has Expired, Completion of the Second CPU Cycle.

**12** The Next User Program Cycle *Prg 1* Starts.

**13** Completion of *Prg 1 Max. Duration for Each Cycle [µs]*, *Prg 2* is Resumed.

**14** Completion of *Prg 2 Max. Duration for Each Cycle [µs]*, *Prg 3* is Resumed.

**15** Completion of the *Prg 3* cycle, *Prg 4* is Resumed. The Remaining Duration is Added to *Prg 4* (Highest Priority *z*).

**16** *Max. Duration for Each Cycle [µs]* of *Prg 4*+ Remaining Duration of *Prg 3* Expired, Completion of the Third CPU Cycle.

Figure 5: Multitasking Mode 2

i The execution time not used by user programs that were not run, is not available as residual time for other user programs. User programs are not run if they are in one of the following states:

- STOP
- ERROR
- TEST_MODE

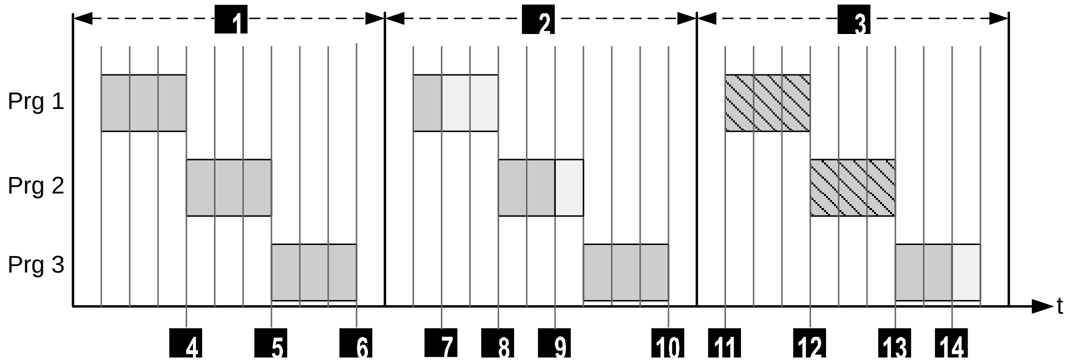
As a consequence, the number of CPU cycles required to process another user program cycle could increase.

**In such a case, if the value set for *Maximum Cycle Count* is too low, the maximum time for processing a user program can be exceeded and result in an error stop!**
**Maximum processing time = *Max. Duration for Each Cycle [µs] * Maximum Number of Cycles***

Use multitasking mode 3 to verify the parameter setting!

3. **Multitasking Mode 3** does not use the unneeded duration for running the user programs, rather, it waits until the *Max. Duration for Each Cycle [µs]* of the user program is reached and then starts processing the next user program. This behavior results in CPU cycles of the same duration.
   Multitasking mode 3 allows users to verify if multitasking mode 2 ensures proper program execution, even in the worst case scenario.

   The example examines user programs named *Prg 1*, *Prg 2* and *Prg 3*:




1 First CPU Cycle Considered.

2 Second CPU Cycle Considered.

3 Third CPU Cycle Considered.

4 *Max. Duration for Each Cycle [µs]* of *Prg 1* has Expired, *Prg 2* Starts.

5 *Max. Duration for Each Cycle [µs]* of *Prg 2* has Expired, *Prg 3* Starts.

6 *Max. Duration for Each Cycle [µs]* of *Prg 3* has Expired, Completion of the First CPU Cycle. *Prg 1* is Resumed.

7 Completion of the User Program Cycle *Prg 1*. Waiting for the Remaining Duration.

8 *Max. Duration for Each Cycle [µs]* of *Prg 1* has Expired. *Prg 2* is Resumed.

9 Completion of the User Program Cycle *Prg 2*. Waiting for the Remaining Duration.

10 *Max. Duration for Each Cycle [µs]* of *Prg 3* has expired. Completion of the Second CPU Cycle.

11 The Next User Program Cycle *Prg 1* Starts.

12 *Max. Duration for Each Cycle [µs]* of *Prog 1* has Expired. The Next User Program Cycle of *Prog 2* Starts.

13 *Max. Duration for Each Cycle [µs]* of *UP 2* has Expired. *Prg 3* is Resumed.

14 Completion of the User Program Cycle *Prg 3*. Standby Time until the *Max. Duration for Each Cycle [µs*] of *Prg 3* has Expired. Completion of the Third CPU Cycle.

Figure 6: Multitasking Mode 3

i In the examples illustrating the multitasking modes, input and output processing are represented as empty spaces at the beginning and the end of each CPU cycle.

## 6.3 Reload

If user programs were modified, the changes can be transferred to the PES during operation. The operating system checks and activates the modified user program which then assumes the control task.

i **Observe the following points when reloading step chains:**

The reload information for step sequences does not take the current sequence status into account. The step sequence can be accordingly changed and set to an undefined state by performing a reload.
The user is responsible for this action.

Examples:

- Deleting the active step. As a result, no step of the step chain has the *active* state.
- Renaming the initial step while another step is active.
  As a result, a step chain has two active steps!

i **Observe the following points when reloading actions:**

During the reload, actions are loaded with their corresponding data. All potential consequences must be carefully analyzed prior to performing a reload.

Examples:

- If a timer action qualifier is deleted due to the reload, the timer expires immediately. Depending on the remaining settings, the *Q* outputs can therefore be set to TRUE.
- If the status action qualifier (e.g., the *S* action qualifier) is deleted for a set element, the element remains set.
- Deleting a *P0* action qualifier set to TRUE actuates the trigger function.

i When reloading the watchdog time, the following points must be observed:

If a reload process is to increase the watchdog time, first set the new value for the watchdog time online and then perform the reload. This avoids connection losses and the shutdown of the I/O module.

Prior to performing a reload, the operating system checks if the required additional tasks would increase the cycle time of the current user programs to such an extent that the defined watchdog time is exceeded. In this case, the reload process is aborted with an error message and the controller continues operation with the previous project configuration.

i **The controller can interrupt a running reload process.**

A successful reload is ensured by planning a sufficient reserve for the reload when determining the watchdog time or temporarily increasing the controller watchdog time by a reserve.

Any temporary increases in the watchdog time must be agreed upon with the responsible test authority.

Also exceeding the target cycle time can result in a reload interruption.

The reload can only be performed if the *Reload Allowed* system parameter is set to ON and the *Reload Deactivation* system variable is set to OFF.

i The user is responsible for ensuring that the watchdog time includes a sufficient reserve time. This should allow the user to manage the following situations:

- Variations in the user program's cycle time
- Sudden, strong cycle loads, e.g., due to communication.
- Expiration of time limits during communication.

During a reload, the global and local variables obtain the values of the corresponding variables from the previous project version. Names of local variables contain the POU instance names.

If names are changed and loaded into the PES by performing a reload, this procedure has the following consequences:

- Renaming a variable has the same effect as deleting the variable and creating a new one, i.e., it results in an initialization process. This also applies to retain variables. The variables lose their current value.
- Renaming a function block instance results in initializing all the variables, including retain variables, and all the function block instances.
- Renaming a program results in initializing all its variables and function block instances.

**This behavior may have unintended effects on one or multiple user programs and therefore on the plant to be controlled!**

## 6.3.1 Conditions for Using the Reload Function

A license is required to use the reload feature.

A reload can be performed to load the following project changes to the controller:

- Changes to the user program parameters.
- Changes to the logic of the program, function blocks and functions.
- Changes that allow a reload in accordance with Table 19.

| Changes to | Type of change | | | |
|---|---|---|---|---|
| | Add | Delete | Change of the initial value | Assignment of other variables |
| Assigning global variables to | | | | |
| User programs | • | • | • | • |
| System variables | • | • | • | • |
| I/O channels | • | • | • | • |
| Communication protocols | • | • | - | - |
| safe**ethernet** | • | • | • | - |
| SOE | • | • | | |
| Communication protocols | • | • | n.a. | n.a. |
| User programs | • | •** | n.a. | n.a. |
| System ID, rack ID | - | | | |
| IP addresses | • | | | |
| User accounts and licenses | • | | | |
| • Reload possible<br>- Reload not possible<br>** Reload possible, but the controller must still contain at least one user program<br>n.a.: not applicable | | | | |

Table 19: Reloading after Changes

A reload may only be performed in accordance with the conditions mentioned in the previous section. In all the other cases, stop the controller and perform a download.

**TIP** Proceed as described below to perform a reload even if global variable assignments have been added:

- While creating the user program, assign unused global variables to communication protocols.
- Assign safe value as initial value to unused global variables.

To a later time point, this assignment must only be changed and not added ensuring the possibility to perform a reload.

## 6.4 Forcing

Forcing is the procedure by which a variable's current value is replaced with a force value. The current value of a variable is assigned from one of the following sources:

- a physical input
- communication
- a logic operation

When a variable is being forced, its value is defined by the user.

Forcing is used for the following purposes:

- Testing the user program; in particular, in cases or conditions that cannot otherwise be tested.
- Simulating unavailable sensors when the initial values are not appropriate.

**WARNING**

**Failure of safety-related operation possible due to forced values!**

- **Forced value may lead to incorrect output values.**
- **Forcing prolongates the cycle time. This can cause the watchdog time to be exceeded.**
- **Forcing is only permitted after receiving consent from the competent test authority.**

When forcing values, the person in charge must take further technical and organizational measures to ensure that the process is sufficiently monitored in terms of safety. HIMA recommends setting a time limit for the forcing procedure, see below.

Forcing can operate at two levels:

- Global forcing: Global variables are forced for all applications.
- Local forcing: Values of local variables are forced for an individual user program.

To force a global or local variable, the following conditions must be met:

- The corresponding force switch is set.
- Forcing was started.

If forcing was started, a change to the force switch has an immediate effect.

If forcing was started and the force switch is set, a change to the force value has an immediate effect.

Local forcing can be started and stopped individually for each user program.

### 6.4.1 Time Limits

Different time limits can be set for global or local forcing. Once the defined time has expired, the controller stops forcing values.

It is possible to define how the HIMatrix system should behave upon expiration of the time limit:

- If global forcing is used, the following settings can be selected:
  - The resource stops.
  - The resource continues to operate.
- If local forcing is used, the following settings can be selected:
  - The user program stops.
  - The user program continues to run.

It is also possible to use forcing without time limit. In this case, the forcing procedure must be stopped manually.

If a variable is no longer forced, the process value is used again for the variable.

### 6.4.2 Force Editor

The SILworX Force Editor displays all the variables for which forcing is allowed. Global and local variables are grouped into two different tabs.

Use these tabs to configure the force values and set the force switches.

### 6.4.3 Restricting the Use of Forcing

The following measures limiting the use of forcing can be configured to avoid potential faults in the safety functionality due to its improper use:

- Configuring different user profiles with or without forcing authorization.
- Prohibit global forcing for a resource.
- Prohibit local forcing or entering new process values.
- Forcing can also be stopped immediately using a key switch.
  To do so, the *Force Deactivation* system variable must be linked to a digital input connected to a key switch.

  The system variable *Force Deactivation* prevents global and local forcing from being started and stops an on-going forcing process.

# 7 Start-Up

Commissioning of modular HIMatrix M45 system comprises the following phases:

- Mounting the DIN rail in a suitable location and its assembly with modules.
  Take the dissipation of the generated heat into account.
- Electrical connection of power supply, earthing, sensors, and actuators
- Configuration
  - Writing the user program
  - Definition of safety, communication and other parameters

## 7.1 Considerations about Heat

The increased integration level of electronic components causes a corresponding lost heat. This depends on the external load of HIMatrix modules. Depending on the structure, the system installation and ventilation are thus of importance.

During the system installation, it must be ensured that the approved environmental requirements have been adhered to. Lowering the operating temperature increases the lifetime and the reliability of the electronic components within the systems.

### 7.1.1 Heat Dissipation

A closed enclosure must be designed such that the heat generated inside can be dissipated through its surface.

Mounting type and position must be chosen such that heat dissipation is ensured.

The power dissipation of the installed equipment is decisive for determining the fan components. It is assumed that heat load and unhindered natural convection are uniformly distributed (see Chapter 7.1.1.3).

#### 7.1.1.1 Definitions

$P_V$ [W] Power dissipation (heat capacity) of the electronic components within the enclosure

A [$m^2$] Effective enclosure surface area (see Table 20)

k [$W/m^2$ K] Heat transfer coefficient of the enclosure, steel sheet: ~ 5.5 $W/m^2$ K

#### 7.1.1.2 Installation Type

In accordance with the mounting or installation type, the effective enclosure surface area A is determined as follows:

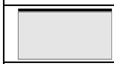
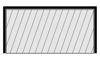
| Enclosure installation type in accordance with VDE 0660, Part 5 | | Calculation of A [$m^2$] |
|---|---|---|
|  | Individual enclosure, free-standing on all sides | A = 1.8 x H x (W + D) + 1.4 x W x D |
|  | Individual enclosure for wall mounting | A = 1.4 x W x (H + D) + 1.8 x H x D |
|  | First or last enclosure in a suite, free-standing | A = 1.4 x W x (H + D) + 1.8 x H x D |
|  | First or last enclosure in a suite, for wall mounting | A = 1.4 x H x (W + D) + 1.4 x W x D |
|  | Central enclosure, free-standing | A = 1.8 x W x H+ 1.4 x W x D + H x D |
|  | Central enclosure for wall mounting | A = 1.4 x W x (H + D) + H x D |
|  | Enclosure within a suite, for wall mounting, covered roof surface | A = 1.4 x W + H+ 0.7 x W x D + H + D |

Table 20: Installation Type

### 7.1.1.3 Natural Convection

When the natural convection is used, the lost heat is dissipated through the enclosure walls. Requirement: The ambient temperature must be lower that the temperature within the enclosure.

The maximum temperature increase $(\Delta T)_{max}$ of all electronic devices within the enclosure is calculated as follows:

$$(\Delta T)_{max} = \frac{P_V}{k \cdot A}$$

The power dissipation $P_V$ can be calculated based on the specifications for the electric power rating of the system and its inputs and outputs.

The temperature within an enclosure can also be calculated in accordance with VDE 0660, Part 507 (HD 528 S2).

i All considerations about heat must take **every** component within a cabinet or enclosure into account!

## 7.2 Installation and Mounting

The safety-related HIMatrix M45 controller systems can be installed on mounting surfaces, and also in closed enclosures such as control stations, terminal boxes and control cabinets. They have been developed in compliance with the relevant standards for EMC, climate and environmental requirements.

The list of the standards that must be observed is specified in Chapter 2.2.1 and in the manuals for the HIMatrix M45 systems.

The protection class of the HIMatrix M45 systems (IP20) can be increased by installing them in an appropriate enclosure in accordance with the requirements. In doing so, appropriate heat dissipation must be ensured (see Chapter 7.1).

### 7.2.1 Mounting

The location for installing a HIMatrix M45 system must be chosen observing the ambient conditions (see Chapter 2.2) to ensure a smooth operation.

Observe the following points:

- For efficient cooling, mount the DIN rail horizontally.
- Maintain a distance of at least 100 mm above and below the M45 system.
- Do not mount the M45 system above heating equipment or any heat source.

Only mount the controller with unconnected connectors.

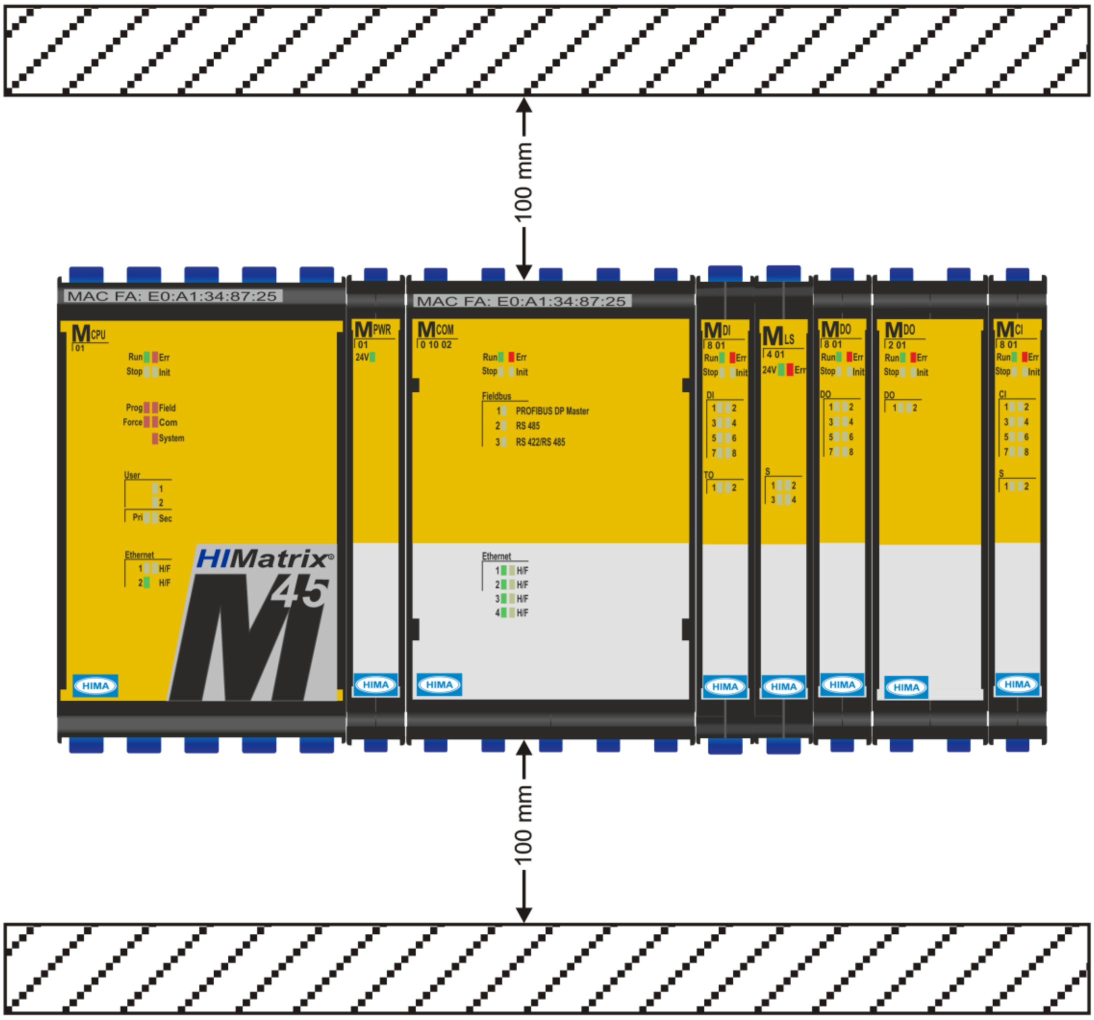



Figure 7: Mounting Distance

---

i The HIMatrix systems must be mounted such that:

- HIMatrix M45 systems are not heated up by other devices with high heat emissions.
- Devices with high EMC interference do not interfere with HIMatrix M45 systems. Observe the instructions provided by the manufacturer.

---

## 7.2.2 Construction Depth

Due to the connections for communication and I/O level, the HIMatrix M45 system has a construction depth of 90 mm. It is measured from the mounting rail.

## 7.2.3 Mounting on a Flat Base

The DIN rail must be secured on a flat base.

## 7.3 Mounting and Removing Modules and Sockets

This chapter describes how to mount and remove the modules and sockets. When replacing modules, the sockets remain on the DIN rail. This saves additional wiring effort since all field lines are connected to the socket.

### 7.3.1 Mounting the Socket

Tools and utilities:

- Screwdriver, slotted 1.0 x 5.5 mm

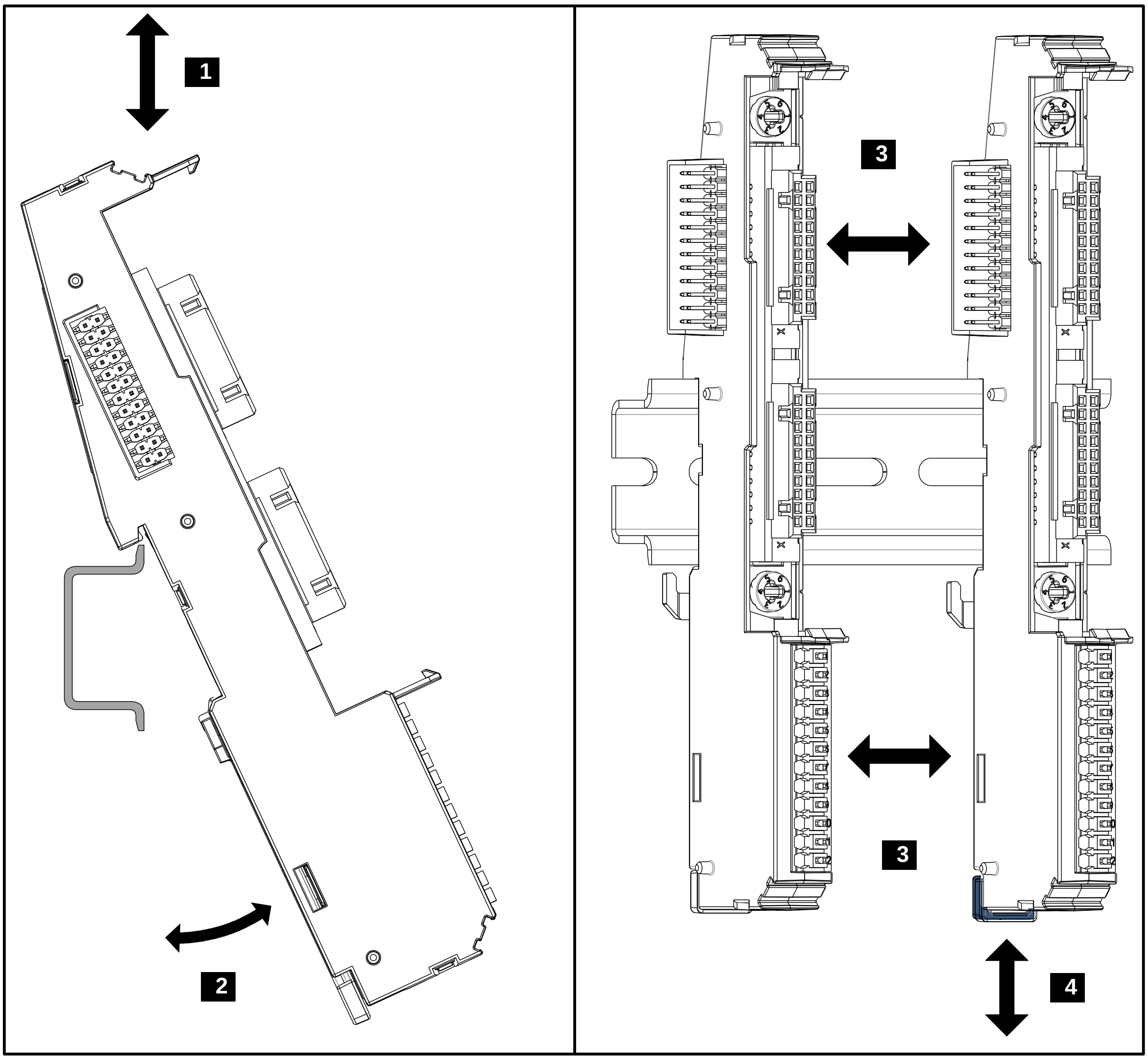
**To insert the socket**

1. Set the socket onto the DIN rail **1**.
2. Swivel the socket in **2**.
3. Move the socket on the DIN rail and connect it to another socket **3**.
4. Press the socket's latch upwards **4**.
   ☑ The latch is used to attach the socket to the DIN rail, and is secured to the socket located on its left-hand side.
5. The socket mounting is completed, the field lines can be connected.

**To remove the socket**

Prior to removing the socket, the module must be removed and the field lines must be released from the terminals.

1. Use a screwdriver to push the blue latch downwards **4**.
2. Remove the sockets from the adjacent sockets **3**.
3. Swivel the socket out **2**.
4. Lift the socket and remove it **1**.




**1** Setting and Lifting the Socket
**2** Swiveling the Socket In and Out
**3** Connecting and Disconnecting Sockets
**4** Closing and Opening the Latch

Figure 8: Example of Socket Mounting

### 7.3.2 Inserting and Removing the Module

This chapter describes how to mount and remove a module in the M45 system.

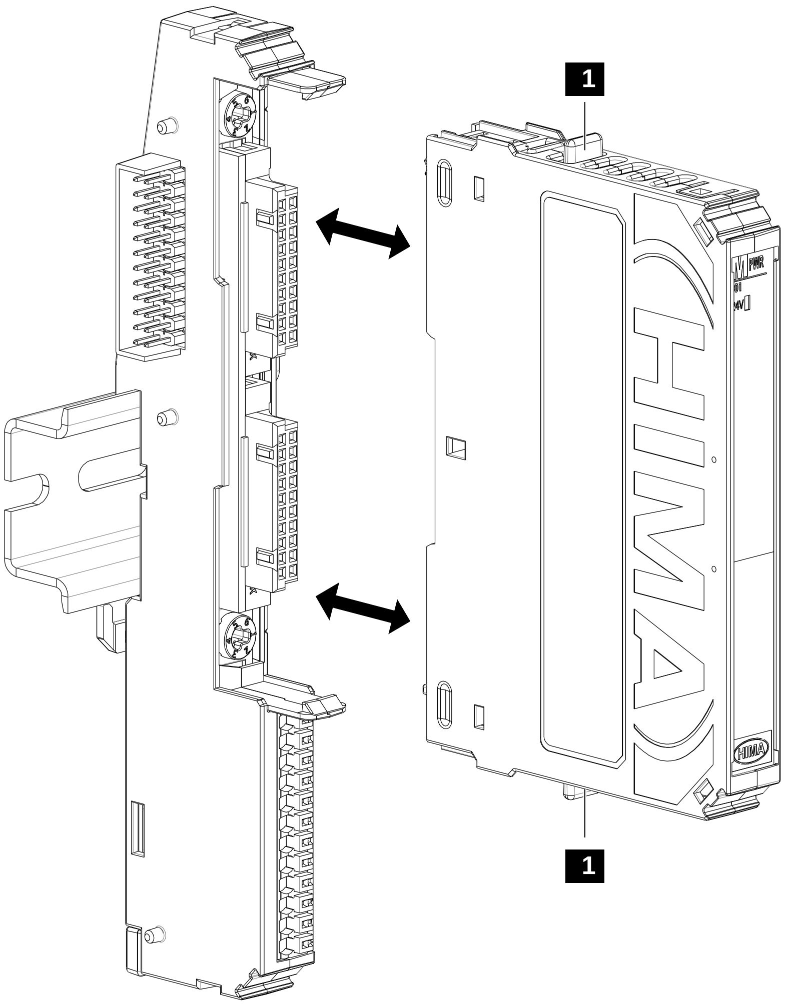
Coding prevents the module from improper assembling.

**To insert the module**

1. Plug the module in to the socket, until the locking mechanism is engaged.

**To remove the module**

1. Press the latch 1 backwards as far as it can go. The locking mechanism is released.
2. Remove the module from the socket.




1 Latch for Releasing the Module

Figure 9: Example of Mounting and Removing the Module

### 7.3.3 Changes to M45 System

Changes to the number of modules, i.e., if modules and sockets are added or removed, may only be performed **if the supply voltage is switched off**!

Changes performed to the HIMatrix M45 system without switching off the supply voltage result in an address conflict.

Address conflicts can be repaired as follows:

- Switching the supply voltage off and on again.
- Restart the concerned modules.
  To do so, restart the modules from left to right, towards increasing slot numbers.
- Restart all modules.

## 7.4 Connecting the Input and Output Circuits

Only personnel with knowledge of ESD protective measures may modify or extend the system wiring.

**NOTE**

**Electrostatic discharge!**

**Failure to comply with these instructions can damage the electronic components.**

- **Prior to working with HIMA components, touch an earthed object.**
- **Make sure that the workspace is free of static and wear an ESD wrist strap.**
- **If not used, ensure that the module is protected from electrostatic discharge, e.g., by storing it in its packaging.**

**NOTE**

**Cable failure is possible under tensile load!**

**Do not use the shield clamp as a strain relief for the connected cable!**

### 7.4.1 Earthing and Shielding

#### 7.4.1.1 Earthing the 24 VDC System Voltage

All HIMatrix M45 systems must be operated with power supply units that comply with the SELV (Safety Extra Low Voltage) or PELV (Protective Extra Low Voltage) requirements. A functional earth is prescribed to improve the electromagnetic compatibility (EMC).

All HIMatrix M45 systems can be operated with earthed L- or unearthed.

##### Unearthed Operation

Unearthed operation offers some advantages in terms of EMC behavior.

Some applications have specific requirements for the unearthed operation of controllers, e.g., in accordance with the EN 50156 standard, earth fault monitoring is required for unearthed operation.

##### Earthed Operation

The earth must be designed in accordance with the standard and must have a separate earth contact through which no power-dependent interference current may flow. Only earthing of negative pole (L-) is permitted. Earthing of positive pole (L+) is not permitted since a potential earth fault on the transmitter line would bypass the affected transmitter.

L- can only be earthed on one place within the system. L- is usually earthed directly behind the power supply unit, e.g., on the busbar. The earthing should be easily accessible and well separable. The earthing resistance must be ≤ 2 Ω.

#### 7.4.1.2 Earthing Connections

The modules are earthed using the DIN rail. The DIN rail must be connected to the earthing connector of the control cabinet or enclosure.

These measures ensure a reliable earth ground and compliance with the current EMC requirements for HIMatrix M45 systems.

#### 7.4.1.3 Shielding

The shielding must be connected to the HIMatrix M45 system and the sensor or actuator housing and earthed on one end to the HIMatrix M45 system side to form a Faraday cage.

In all other devices, the shielding must be positioned in the controller housing, terminal box, control cabinet, etc.

i The shield clamp must not be used as a strain relief for the connected cable.

#### 7.4.1.4 EMC Protection

Windows in the enclosure in which the HIMatrix M45 system is installed are permitted.

Increased EMC interferences outside the standard limit values require appropriate measures.

i
- For improved EMC, earth the housing.
- The connection to the next grounding point must be as short as possible to achieve a low earthing resistance.

### 7.4.2 Connecting the Supply Voltage

The electrical connection is made via a removable connector on the processor module socket and on additional power module sockets. The connector can accept wires of up to 6 mm².

| Connection | Function |
|---|---|
| L+ DC 24 V | Power supply L+ (24 VDC) |
| L- DC 24 V | Power supply L- (24 VDC ground) |

Table 21: Connectors for Operating Voltage

If shielded wires are used for the power supply, the shielding must also be connected to the power supply connector via the earth contact.

The power supply unit must meet the IEC/EN 61131-2 requirements and comply with the low voltage directives SELV (Safety Extra Low Voltage) or PELV (Protective Extra Low Voltage).

Check proper polarity, voltage and ripple prior to connecting the operating voltage 24 VDC.

**Do not exchange the terminals L+ and L-, or connect them to other terminals of the device!**

In case of a false connection, a pre-fuse blows to prevent the device from being damaged.

## 7.5 Configuration

This chapter describes the configuration using the SILworX programming tool.

### 7.5.1 Configuring the Resource

The resource properties and the hardware output variables are changed at this level.

### 7.5.1.1 Resource Properties

The system parameters of the resource determine the controller behavior during operation and can be set in SILworX, in the *Properties* dialog box of the resource.

| Parameter | Description | | Default value | Setting for safe operation |
|---|---|---|---|---|
| Name | Name of the resource | | | Arbitrary |
| **System ID [SRS]**[1)] | System ID of the resource<br>1...65 535<br>The value assigned to the system ID must differ to the default value, otherwise the project is not able to run! | | 60 000 | Unique value within the controller network. This network includes all controllers that can potentially be interconnected |
| **Safety Time [ms]** | Safety time in milliseconds<br>20...22 500 ms | | 600 ms | Application-specific |
| **Watchdog Time [ms]** | Watchdog time in milliseconds: 4...5000 ms | | 200 ms | Application-specific |
| Target Cycle Time [ms] | Targeted or maximum cycle time, see *Target Cycle Time Mode*, 0...7500 ms. The maximum target cycle time value may not exceed the *watchdog time* - minimum watchdog time; otherwise it is rejected by the PES.<br>If the default value 0 ms is set, the target cycle time is not taken into account. | | 0 ms | Application-specific |
| Target Cycle Time Mode | Use of *Target Cycle Time [ms]* see Table 23. | | Fixed-tolerant | Application-specific |
| Multitasking Mode | Mode 1 | The duration of a CPU cycle is based on the required execution time of all user programs. | Mode 1 | Application-specific |
| | Mode 2 | The processor provides user programs with a higher priority the execution time not needed by user programs with a lower priority. Operation mode for high availability. | | |
| | Mode 3 | The processor waits until the execution time not needed by the user programs has expired, thus increasing the cycle. | | |
| Max.Com. Time Slice ASYNC [ms] | Highest value in ms for the time slice used for communication during a resource cycle, see the communication manual (HI 801 101 E),<br>2...5000 ms | | 60 ms | Application-specific |
| Max. Duration of Configuration Connections [ms] | It defines how much time within a CPU cycle is available for configuration connections, 2...3500 ms | | 12 ms | Application-specific |
| Maximum System Bus Latency [µs] | Not applicable for HIMatrix M45 controllers! | | 0 µs | - |

| Parameter | Description | Default value | Setting for safe operation |
|---|---|---|---|
| **Allow Online Settings** | ON: **All** the switches/parameters listed below OFF can be changed online using the PADT. This is only valid if the system variable *Read-only in RUN* has the value OFF.<br>OFF: These parameters may **not** be changed online<br>▪ *System ID*<br>▪ *Autostart*<br>▪ *Global Forcing Allowed*<br>▪ *Global Force Timeout Reaction*<br>▪ *Load Allowed*<br>▪ *Reload Allowed*<br>▪ *Start Allowed*<br>These parameters may be changed online if *Reload Allowed* is set to ON.<br>▪ Watchdog Time (for the resource)<br>▪ *Safety Time*<br>▪ *Target cycle time*<br>▪ *Target Cycle Time Mode*<br>If *Reload Allowed* is set to OFF, they are not changeable online.<br>i *Allow Online Settings* can only be set to ON via reload or if the PES is stopped. | ON | OFF is recommended |
| **Autostart** | ON: If the processor module is connected to the supply voltage, the user program/programs start automatically<br>OFF: The user program does not start automatically after connecting the supply voltage. | OFF | Application-specific |
| **Start Allowed** | ON: A cold start or warm start permitted with the PADT in RUN or STOP<br>OFF: Start not allowed | ON | Application-specific |
| **Load Allowed** | ON: Configuration download is allowed<br>OFF: Configuration download is not allowed | ON | Application-specific |
| **Reload Allowed** | ON: Configuration reload is allowed<br>OFF: Configuration reload is not allowed.<br>A running reload process is not aborted when switching to OFF | ON | Application-specific |
| **Global Forcing Allowed** | ON: Global forcing permitted for this resource<br>OFF: Global forcing not permitted for this resource | ON | Application-specific |
| Global Force Timeout Reaction | Specifies how the resource should behave when the global force timeout has expired:<br>▪ Stop Forcing<br>▪ Stop Resource | Stop Forcing | Application-specific |
| Minimum Configuration Version | With this setting, code compatible to previous or newer CPU operating system versions in accordance with the project requirements may be generated.<br>SILworX V2, SILworX V3, SILworX V4, SILworX V5: Not applicable for HIMatrix M45 controllers!<br>SILworX V6: Setting for HIMatrix M45. Generates code suitable for the CPU operating system V9. This setting ensures the compatibility with future versions. | SILworX V6 | SILworX V6 |
| [1] **Safety parameters** are displayed in bold. | | | |

Table 22: Resource System Parameters

The following table describes the effect of *Target Cycle Time Mode*.

| Target Cycle Time Mode | Effect on user programs | Effect on reload of processor modules |
|---|---|---|
| Fixed | The PES maintains the target cycle time and extends the cycle if necessary. If the processing time of the user programs exceeds the target cycle time, the cycle duration is increased. | Reload is not processed if the target cycle time is not sufficient. |
| Fixed-tolerant | | At most, the duration of every fourth cycle is increased to allow reload. |
| Dynamic-tolerant | HIMatrix executes the cycle as quickly as possible. | At most, the duration of every fourth cycle is increased to allow reload. |
| Dynamic | | Reload is not processed if the target cycle time is not sufficient. |

Table 23: Effect of Target Cycle Time Mode

### 7.5.1.2 Hardware System Variables for Setting the Parameters

These variables are used to change the behavior of the controller while it is operating in specific states. These variables are located in the hardware detail view of the SILworX Hardware Editor.

| Variable | Function | Default setting | Setting for safe operation |
|---|---|---|---|
| Force Deactivation | Used to prevent forcing and to stop it immediately | FALSE | Application-specific |
| Spare 2... Spare 16 | No function | - | - |
| Emergency stop 1...<br>Emergency stop 4 | Emergency stop switch to shut down the controller if faults are detected by the user program | FALSE | Application-specific |
| Read-only in RUN | After starting the controller, no operating action such as stop, start or download is permitted in SILworX , except for forcing and reload. | FALSE | Application-specific |
| Relay Contact 1...<br>Relay Contact 4 | Not applicable for M45!<br>It controls the corresponding relay contacts, if existing. | FALSE | Application-specific |
| Reload Deactivation | The controller cannot be loaded by performing a reload. | FALSE | Application-specific |
| Power save mode | It switches the output modules to power save mode, i.e., the outputs are switched off. | FALSE | Application-specific |
| User LED 1...<br>User LED 2 | It controls the corresponding LED. | FALSE | Application-specific |

Table 27: Hardware System Variables

Global variables can be connected to these system variables; the value of the global variables is modified using a physical input or the user program logic.

The system variables *Force Deactivation*, *Read-only in Run* and *Reload Deactivation* can be activated by authorized persons using a key switch.

Using the suitable key switch, the user is thus able to interrupt running processes, e.g., forcing.

**To configure *Force Deactivation*, *Read-only in Run* or *Reload Deactivation* for being operated using a key switch**

1. Assign a global variable to a system variable.
2. Assign the same global variable to a digital input.
3. Connect a key switch to the digital input.

The position of the key switch defines the system variable value.

One key switch can be used to control several system variables.

### 7.5.1.3 Hardware System Variables for Reading the Parameters

These system variables can be accessed in the SILworX Hardware Editor.

Select the gray background outside the (yellow) subrack representation and double-click or use the context menu to open the detail view.

| Variable | Description | Data type |
|---|---|---|
| Number of Field Errors | Number of current I/O errors | UDINT |
| Number of Field Errors - Historic Count | Counted number of I/O errors (counter resettable) | UDINT |
| Number of Field Warnings | Number of current I/O warnings | UDINT |
| Number of Field Warnings - Historic Count | Counted number of I/O warnings (counter resettable) | UDINT |
| Communication Error Count | Number of current communication errors | UDINT |
| Communication Error Historic Count | Counted number of communication errors (counter resettable) | UDINT |
| Communication Warning Count | Number of current communication warnings | UDINT |
| Communication Warnings Historic Count | Counted number of communication warnings (counter resettable) | UDINT |
| System Error Count | Number of current system errors | UDINT |
| System Error Historic Count | Counted number of system errors (counter resettable) | UDINT |
| System Warning Count | Number of current system warnings | UDINT |
| System Warning Historic Count | Counted number of system warnings (counter resettable) | UDINT |
| Autostart | ON: When the processor module is connected to the supply voltage, it automatically starts the user program. | BOOL |
| | OFF: When the supply voltage is connected, the processor module enters the STOP state. | |
| OS Major | Operating system version contained in the processor module | UINT |
| OS Minor | | UINT |
| CRC | Resource configuration checksum | UDINT |
| Date/time [ms portion] | System date and time in s and ms since 1970-01-01 | UDINT |
| Date/time [s portion] | | UDINT |
| Force Deactivation | ON: Forcing is deactivated. | BOOL |
| | OFF: Forcing is possible. | |
| Forcing Active | ON: Global or local forcing is active. | BOOL |
| | OFF: Global and local forcing are not active. | |
| Force Switch State | State of the force switch | UDINT |
| | 0xFFFFFFFE No force switch set | |
| | 0xFFFFFFFF At least one force switch set | |
| Global Forcing Started | ON: Global forcing is active. | BOOL |
| | OFF: Global forcing is not active. | |
| Spare 0...Spare 16 | Reserved | USINT |
| Spare 17 | | BOOL |
| Last Field Warning [ms] | Date and time of the last I/O warning in s and ms since 1970-01-01 | UDINT |
| Last Field Warning [s] | | UDINT |

| Variable | Description | | Data type |
|---|---|---|---|
| Last Communication Warning [ms] | Date and time of the last communication warning in s and ms since 1970-01-01 | | UDINT |
| Last Communication Warning [s] | | | UDINT |
| Last System Warning [ms] | Date and time of the last system warning in s and ms since 1970-01-01 | | UDINT |
| Last System Warning [s] | | | UDINT |
| Last Field Error [ms] | Date and time of the last I/O error in s and ms since 1970-01-01 | | UDINT |
| Last Field Error [s] | | | UDINT |
| Last Communication Error [ms] | Date and time of the last communication error in s and ms since 1970-01-01 | | UDINT |
| Last Communication Error [s] | | | UDINT |
| Last System Error [ms] | Date and time of the last system error in s and ms since 1970-01-01 | | UDINT |
| Last System Fault [s] | | | UDINT |
| Fan State | 0x00 | Fan functioning | BYTE |
| | 0x01 | Fan defective | |
| | 0xFF | Not available - for M45 | |
| Allow Online Settings | Specifies if the online settings of the enable switch are allowed: | | BOOL |
| | ON: | The subordinate enable switches can be changed online. | |
| | OFF: | The subordinate enable switches cannot be changed online. | |
| Read-only in RUN | ON: | The operator actions: Stop, Start, Download are locked. | BOOL |
| | OFF: | The operator actions: Stop, Start, Download are not locked. | |
| Reload Allowed | ON: | Controller can be loaded by performing a reload | BOOL |
| | OFF: | Controller cannot be loaded by performing a reload | |
| Reload Deactivation | ON: | Loading by performing a reload is locked | BOOL |
| | OFF: | Loading by performing a reload is possible | |
| Reload Cycle | TRUE during the first cycle after a reload, otherwise FALSE | | BOOL |
| Safety Time [ms] | Safety time set for the controller in ms | | UDINT |
| Start Allowed | ON: | Start of processor module using PADT is allowed | BOOL |
| | OFF: | Start of processor module using PADT is not allowed | |
| Start Cycle | ON during the first cycle after starting, otherwise OFF | | BOOL |

<table>
<tr><th>Variable</th><th>Description</th><th>Data type</th></tr>
<tr><td>Power Supply State</td><td>Bit-coded state of the power supply units.<br><table><tr><th>Value</th><th>State</th></tr><tr><td>0x00</td><td>Normal</td></tr><tr><td>0x01</td><td>Undervoltage with 24 V supply voltage</td></tr><tr><td>0x02</td><td>(Low voltage with battery) not used</td></tr><tr><td>0x04</td><td>Low voltage with internally generated 5 V voltage</td></tr><tr><td>0x08</td><td>Low voltage with internally generated 3.3 V voltage</td></tr><tr><td>0x10</td><td>Overvoltage with internally generated 3.3 V voltage</td></tr></table>F60 modular controller:<br><table><tr><th>Value</th><th>State</th></tr><tr><td>0x00</td><td>Normal</td></tr><tr><td>0x01</td><td>Error with 24 V supply voltage</td></tr><tr><td>0x02</td><td>Battery failure</td></tr><tr><td>0x04</td><td>Error with 5 V voltage of power supply</td></tr><tr><td>0x08</td><td>Error with 3.3 V voltage of power supply</td></tr><tr><td>0x10</td><td>Undervoltage with 5 V voltage</td></tr><tr><td>0x20</td><td>Overvoltage with 5 V voltage</td></tr><tr><td>0x40</td><td>Undervoltage with 3.3 V voltage</td></tr><tr><td>0x80</td><td>Overvoltage with 3.3 V voltage</td></tr></table></td><td>BYTE</td></tr>
<tr><td>System ID [SRS]</td><td>System ID of the controller, 1...65 535</td><td>UINT</td></tr>
<tr><td>Systemtick HIGH</td><td rowspan="2">Revolving millisecond counter (64-bit)</td><td>UDINT</td></tr>
<tr><td>Systemtick LOW</td><td>UDINT</td></tr>
<tr><td>Temperature state</td><td>Bit-coded temperature state of the processor module<br><table><tr><th>Value</th><th>State</th></tr><tr><td>0x00</td><td>Normal temperature</td></tr><tr><td>0x01</td><td>Temperature threshold 1 exceeded</td></tr><tr><td>0x03</td><td>Temperature threshold 2 exceeded</td></tr><tr><td>0xFF</td><td>Not available</td></tr></table></td><td>BYTE</td></tr>
<tr><td>Remaining Global Force Duration [ms]</td><td>Time in ms until the time limit set for global forcing expires.</td><td>DINT</td></tr>
<tr><td>Watchdog Time [ms]</td><td>Maximum permissible duration of a RUN cycle in ms.</td><td>UDINT</td></tr>
<tr><td>Cycle Time, last [ms]</td><td>Current cycle time in ms</td><td>UDINT</td></tr>
<tr><td>Cycle Time, max [ms]</td><td>Maximum cycle time in ms</td><td>UDINT</td></tr>
<tr><td>Cycle Time, min [ms]</td><td>Minimum cycle time in ms</td><td>UDINT</td></tr>
<tr><td>Cycle Time, average [ms]</td><td>Average cycle time in ms</td><td>UDINT</td></tr>
</table>

Table 28: Hardware System Variables for Reading the Parameters

### 7.5.1.4 Rack System Parameter Settings

The rack system parameters can be set in the Rack's online view.

| Parameter | Description | Default value |
|---|---|---|
| Type | Spare, not changeable | - |
| Name | Rack name for the controller, text | HIMatrix F.. Rack |
| Rack ID | Not changeable | 0 |
| Temperature monitoring | Specifies the temperature threshold that, if exceeded, causes a warning message to be created:<br>▪ Warning at temperature thresholds 1 and 2<br>▪ Warning at temperature threshold 2 only<br>▪ Warning at temperature threshold 1 only<br>▪ No warning at temperature thresholds | Warning at temperature thresholds 1 and 2 |

Table 29: System Parameters of the Rack

## 7.5.2 Configuring the Ethernet Interfaces

Ethernet interfaces are configured in the Detail View of the communication module. If the remote I/Os have no communication module, the Ethernet interfaces are configured in the Detail View of the processor module. Refer to the manuals of the HIMatrix controllers and remote I/Os for more details.

## 7.5.3 Configuring the User Program

The following user program switches and parameters can be set in the *Properties* dialog box of the user program:

| Switch / Parameter | Function | Default value | Setting for safe operation |
|---|---|---|---|
| Name | Name of the user program | | Arbitrary |
| Program ID | ID for identifying the program when displayed in SILworX, 0…4 294 967 295. | 0 | Application-specific |
| Priority | Priority of the user program: 0...31, if multitasking is used | 0 | Application-specific |
| Program's Maximum Number of CPU Cycles | Maximum number of CPU cycles that a user program cycle may encompass. | 1 | Application-specific |
| Max. Duration for Each Cycle [µs] | Maximum time in each processor module cycle for executing the user program: 1...4 294 967 295 µs.<br>Set to 0: No limitation. | 0 µs | 0 µs |
| Classification | Classification of the user program: *Safety-related* or *Standard* (for documentation only). | Safety-related | Application-specific |
| Allow Online Settings | It enables changes of other user program switches during operation.<br>It only applies if the *Allowed Online Settings* switch for the resource is set to ON! | ON | - |
| Autostart | Enabled type of Autostart: Cold Start, Warm Start, Off | Warm Start | Application-specific |
| Start Allowed | ON: The PADT may be used to start the user program.<br>OFF: The PADT may not be used to start the user program | ON | Application-specific |
| Test Mode Allowed | ON The test mode is permitted for the user program.<br>OFF The test mode is not permitted for the user program. | OFF | Application-specific |

<table>
<tr><th>Switch / Parameter</th><th colspan="2">Function</th><th>Default value</th><th>Setting for safe operation</th></tr>
<tr><td rowspan="2">Reload Allowed</td><td colspan="2">ON: User program reload is permitted</td><td rowspan="2">ON</td><td rowspan="2">Application-specific</td></tr>
<tr><td colspan="2">OFF: User program reload is not permitted</td></tr>
<tr><td rowspan="2">Local Forcing Allowed</td><td colspan="2">ON: Forcing permitted at program level</td><td rowspan="2">OFF</td><td rowspan="2">OFF is recommended</td></tr>
<tr><td colspan="2">OFF: Forcing not permitted at program level</td></tr>
<tr><td>Local Force Timeout Reaction</td><td colspan="2">Behavior of the user program after the forcing time has expired:<br>▪ Stop Forcing Only.<br>▪ Stop Program.</td><td>Stop Forcing Only.</td><td>Application-specific</td></tr>
<tr><td rowspan="3">Code Generation Compatibility</td><td>SILworX V4</td><td>Code generation is compatible with SILworX V4.</td><td rowspan="3">SILworX V4</td><td rowspan="3">SILworX V4</td></tr>
<tr><td>SILworX V3</td><td>Not applicable for HIMatrix controllers!</td></tr>
<tr><td>SILworX V2</td><td>Code generation is compatible with SILworX V2. Not applicable/not recommended for HIMatrix M45 controllers!</td></tr>
</table>

Table 30: System Parameters of the User Program

Notes specific to the *Code Generation Compatibility* Parameter*:*

- In a new project, SILworX selects the latest value for the *Code Generation Compatibility* parameter. This ensures that the current, enhanced features are activated and the latest hardware and operating system versions are supported. Verify that this setting is in accordance with the hardware in use. HIMatrix M45 needs the value SILworX V4 for the parameter *Minimum Configuration Version*.
- In a project converted from a previous SILworX version, the value for *Code Generation Compatibility* remains the value set in the previous version. This ensures that the configuration CRC resulting from the code generation is the same as in the previous version and the generated configuration is compatible with the operating systems in the hardware.

  For this reason, the value of *Code Generation Compatibility* should not be changed for converted projects.
- If a *Minimum Configuration Version* of *SILworX V4* or higher is set for a resource (see above), the *Code Generation Compatibility* parameter must be set to *SILworX V4* in every user program.

### 7.5.4 Configuration of the Hardware Image

Before configuring the details of inputs, outputs and communication interfaces, create the image of the M45 hardware used in the SILworX Hardware Editor. The image contains all modules in the order they are placed on the DIN rail (or should be placed). The image is created independently of the hardware installation.

The Hardware Editor only represents processor modules, communication modules, and I/O modules. Additional hardware, as the power module M-PWR 01, cannot be represented.

While planning an M45 resource, it is possible to insert modules between adjacent modules. In this case, SILworX shifts all modules located to the right of the insertion position, one position further to the right. If a module is removed, SILworX shifts all modules to its right one position to the left.

The **Insert Linebreak** function can be used to represent the system bus extension and how the modules are distributed over multiple rails. **Insert Linebreak** wraps the modules into multiple rows. The sockets for extending the system bus necessary for the real hardware are not represented.

### 7.5.5 Configuring the Inputs and Outputs

In the Hardware Editor, the inputs and outputs are configured by connecting global variables to the system variables for input and output channels.

**To access the system variables for input and output channels**

1. Display the desired resource in the Hardware Editor.
2. Double-click the required input or output module to open the corresponding detail view.
3. In the detail view, open the tab with the required channels.

The system variables for the channels appear.

#### 7.5.5.1 Use of Digital Input

**To use the value of a digital input in the user program**

1. Define a global variable of type BOOL.
2. Enter an appropriate initial value, when defining the global variable.
3. Assign the global variable to the channel value of the input.
4. In the user program, program a safety-related fault reaction using the status *Channel OK [BOOL]*.

The global variable provides values to the user program.

To get additional options for programming fault reactions in the user program, assign global variables to *Data valid* and *Module OK*. Refer to the module-specific manuals for details on the parameters.

#### 7.5.5.2 Use of Safety-Related Counter Inputs

**To use the integer value**

1. Define a global variable of type UDINT.
2. Enter an appropriate initial value, when defining the global variable.
3. Assign the global variable to the integer value *->Counter Reading[UDINT]* of the input.
4. In the user program, program a safety-related fault reaction using the status *Module OK*.

The global variable provides values to the user program.

To get additional options for programming fault reactions in the user program, assign global variables to *Data valid* and *Module OK*. Refer to the module-specific manual for further details on how to use the parameters of the counter input.

#### 7.5.5.3 Use of Digital Input

**To write a value in the user program to a digital output**

1. Define a global variable of type BOOL containing the value to be output.
2. Enter an appropriate initial value, when defining the global variable.
3. Assign the global variable the *Value [BOOL]* -> channel value of the output.
4. In the user program, program a safety-related fault reaction using the status *Channel OK [BOOL]*.

The global variable provides values to the digital output.

To get additional options for programming fault reactions in the user program, assign global variables to *Data valid* and *Module OK*. Refer to the module-specific manuals for details.

### 7.5.6 Generating the Resource Configuration

With the following procedure, the code is generated twice and the resulting checksums (CRCs) are compared with one another.

**To generate the code for the resource configuration**

1. Select the resource in the structure tree.
2. Click the **Code Generation** button on the Action Bar or select **Code Generation** from the context menu.
   ☑ The *Code Generation <Resource Name>* dialog box appears.

3. Select **CRC Comparison** from the *Code Generation <Resource Name>* dialog box (default value).
4. In the dialog box, click **OK**.
   ☑ An additional *Code Generation <resource name>* appears, shows the progress of the two code generation processes and is closed again. The logbook contains one row informing about the code generation result and one row reporting the result of the CRC comparison.

A valid code is generated for the resource configuration.

Generating the code twice and comparing the CRCs ensures that the non-safe PC did not cause any corruption of the user program.

### 7.5.7 Configuring the System ID and the Connection Parameters

**To configuring the system ID and the connection parameters**

1. Select the resource in the structure tree.
2. Click the **Online** button on the Action Bar or select **Online** from the context menu.
   ☑ The *System Login* dialog box appears.
3. Click **Search**.
   ☑ The *Search per MAC* dialog box appears.
4. Enter the MAC address valid for the controller - see the label on the housing - and click **Search**.
   ☑ In the dialog box, the values set for IP address, subnet mask and S:R:S are displayed.
5. If the values for the project are not correct, click **Change**.
   ☑ The *Write via MAC* dialog box appears.
6. Type correct values for the connection parameters and the SRS, and enter the access data for a user account with administrator rights valid on the controller. Click **Write**.

Connection data and SRS are configured and it is now possible to log in.

For further details, refer to the SILworX first steps manual (HI 801 103 E).

### 7.5.8 Loading a Resource Configuration from the PADT

Before a user program can be loaded with the connection parameters (IP address, subnet mask and system ID) into the controller, the code must have been generated for the resource, and the connection parameters for PADT and resource must be valid, see Chapter 6.9.6.

**To load a resource configuration from the PADT**

1. Select a resource in the structure tree.
2. Click the **Online** button on the Action Bar or select **Online** from the context menu.
3. In the *System Login* dialog box, enter a user group with administrator rights or write access.
   ☑ The Control Panel appears in the workspace and displays the controller state.
4. In the **Online** menu, select **Resource Download**.
   ☑ The *Resource Download* dialog box appears.
5. Click **OK** to confirm the download.
   ☑ SILworX loads the configuration into the controller.
6. Upon completion of the loading procedure, click the **Resource Cold Start** function on the **Online** menu to start the user program.
   ☑ After the cold start, *System State* and *Program Status* enter the RUN state.

The resource configuration is loaded from the PADT.

The Start, Stop and Load functions are also available as symbols on the Symbol Bar.

### 7.5.9 Setting the Date and the Time

**To set the controller's time and date**

1. Select a resource in the structure tree.
2. Click the **Online** button on the Action Bar or select **Online** from the context menu.
3. In the *System Login* dialog box, enter a user group with administrator rights or write access.
   ☑ The Control Panel appears in the workspace and displays the controller state.
4. In the **Online** menu, select **Start-Up** -> **Set Date/Time**
   ☑ The *Set Date/Time* dialog box appears.
5. Select one of the following options:
   - **Use the PADT date and time** - to transfer the time and date displayed for the PADT to the controller.
   - **User-defined** - to transfer the date and time from the two input boxes into the controller. Make sure that the format used for date and time is correct!
6. Click **OK** to confirm the action.

The time and the date are set for the controller.

## 7.6 User Management in SILworX

SILworX can set up and maintain an own user management scheme for each project and controller.

### 7.6.1 User Management for SILworX Projects

A PADT user management scheme for administering the access to the project can be added to every SILworX project.

If no PADT user management scheme exists, any user can open and modify the project. If a user management scheme has been defined for a project, only authorized users can open the project. Only users with the corresponding rights can modify the projects. The following authorization types exist.

| Type | Description |
|---|---|
| Security Administrator (Sec Adm) | Security administrators can modify the user management scheme: setting up, deleting, changing the PADT user management scheme, and the user accounts and user groups, and setting up the default user account.<br>Additionally, they can perform all SILworX functions. |
| Read and Write (R/W) | All SILworX functions, except for the user management |
| Read only (RO) | Read-only access, i.e., the users may not change or archive the projects. |

Table 24: Authorization Types for the PADT User Management Scheme

The user management scheme allocates the rights to the user groups. The user groups allocate the rights to the user accounts assigned to them.

Characteristics of user groups:

- The name must be unique within the project and must contain 1...31 characters.
- A user group is assigned an authorization type.
- A user group may be assigned an arbitrary number of user accounts.
- A project may contain up to 100 user groups.

User account properties

- The name must be unique within the project and must contain 1...31 characters.
- A user account is assigned a user group.

- A project may contain up to 1000 user accounts.
- A user account can be the project default user.

### 7.6.2 User Management for the Controller

The user management scheme for a controller (PES user management) is used to protect the HIMatrix controller against unauthorized access and actions. The users and their access permissions are part of the project; they are defined with SILworX and loaded into the processor module.

The user management scheme allows administrators to define and manage the access permissions for up to ten users. The access permissions are stored in the controller and remain valid after switching off the supply voltage.

Each user account is composed of name, password and access permission. Once the project has been loaded into the controller by performing a download, the user data are available for logging in. The user accounts of a controller can also be used for the corresponding remote I/Os.

Users log in to a controller using their user name and password.

Creating user accounts is not required, but is a contribution to a safe operation. If a user management scheme is defined for a resource, it must contain at least one user with administrator rights.

#### Default User

The factory user settings apply if no user accounts were set up for a resource. The factory settings also apply after starting a controller with the reset pushbutton activated.

Factory settings

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

i Note that the default settings cannot be maintained if new user accounts are defined.

### Parameters for User Accounts

To set up new user accounts, the following parameters must be set:

| Parameter | Description |
|---|---|
| User Name | User name or ID used to log in to a controller.<br>The user name must not contain more than 32 characters (recommended: a maximum of 16 characters) and may only be composed of letters (A ... Z, a ... z), numbers (0 ... 9) and the special characters underscore «_» and hyphen «-».<br>The user name is case sensitive. |
| Password | Password assigned to a user name required for the login.<br>The password must not contain more than 32 characters and may only be composed of letters (A...Z, a...z), numbers (0...9) and the special characters underscore «_» and hyphen «-»<br>The password is case sensitive. |
| Confirm Password | Repeat the password to confirm the entry. |
| Access Mode | The access modes define the permission associated with the user.<br>The following access types are available:<br>▪ Read: Users may only read information but they cannot modify the controller.<br>▪ Read + Operator: Similar to *Read*, but users may also:<br>Perform a download to load and start user programs<br>Reset cycle time and fault or error statistics<br>Set the system time, force, restart and reset modules<br>Start system operation for processor modules.<br>▪ Read + Write: Similar to Read + Operator, but users may also:<br>Load programs into the controller<br>Test programs<br>▪ Administrator: Similar to Read + Write, but users may also:<br>Load operating systems.<br>Modify the main enable switch setting<br>Change the SRS<br>Change the IP settings<br>At least one user must have administrator rights, otherwise the controller settings are not accepted.<br>The administrator can revoke the user's permission to access a controller by deleting the user name from the list. |

Table 25: Parameters for User Accounts in the PES User Management Scheme

### 7.6.3 Setting up User Accounts

A user with administrator rights can access to all user accounts.

Observe the following points when setting up user accounts:

- Make sure that at least one user account is assigned with administrator rights. Define a password for the user account with administrator rights.
- In SILworX, use *Verification* to check the created user account.
- The new user accounts are valid once the code has been generated and a download has been performed to load the project into the controller. All the user accounts previously saved, e.g., the default settings, are no longer valid.

## 7.7 Configuration with SILworX

Depending on the application, the following elements must be configured:

- Ethernet/safe**ethernet**.
- Standard Protocols

Refer to the SILworX communication manual (HI 801 101 E) for more information on how to configure the protocols.

### 7.7.1 Configuring the Ethernet Interfaces

Ethernet interfaces are configured in the Detail View of the communication module.

For HIMatrix systems, set the *Speed Mode [Mbit/s]* and *Flow Control Mode* to **Autoneg** in the Ethernet switch settings.

The parameters *ARP Aging Time*, *MAC Learning*, *IP Forwarding*, *Speed [Mbit/s]* and *Flow Control* are explained in details in the SILworX online help.

---

i Replacement of one controller with identical IP address:

If a controller has its *ARP Aging Time* set to 5 minutes and its *MAC Learning* set to **Conservative**, its communication partner does not adopt the new MAC address until a period of 5 to 10 minutes after the controller is replaced. During this time period, no communication is possible with the replaced controller.

---

The port settings of the integrated Ethernet switch on a HIMatrix resource can be configured individually. In the **Ethernet Switch** tab, an entry can be created for each switch port.

| Name | Explanation |
|---|---|
| Port | Port number as printed on the housing; per port, only one configuration may exist.<br>Range of values: 1...n, depending on the resource |
| Speed [Mbit/s] | 10: Data rate 10 Mbit/s<br>100: Data rate 100 Mbit/s<br>Autoneg: Automatic baud rate setting<br>Standard: Autoneg |
| Flow Control | Full duplex: Simultaneous communication in both directions<br>Half duplex: Communication in both directions, but only one direction at a time<br>Autoneg: Automatic communication control<br>Standard: Autoneg |
| Autoneg also with fixed values | The *Advertising* function (forwarding the Speed and Flow Control properties) is also performed if *Speed* and *Flow Control* have fixed values.<br>This allows other devices with ports set to **Autoneg** to recognize the HIMatrix ports' settings. |
| Limit | Limit the inbound multicast and/or broadcast packets.<br>Off: No limitation<br>Broadcast: Limit broadcast packets (128 kbit/s)<br>Multicast and Broadcast: Limit multicast and broadcast packets (1024 kbit/s)<br>Default: Broadcast |

Table 26: Parameters of the Port Configuration

To modify and enter these parameters in the communication system's configuration, double-click each table cell. The parameters become operative for HIMatrix communication, once they have been re-compiled with the user program and transferred to the controller.

The properties of the communication system and Ethernet switch can also be changed online using the Control Panel. These settings become operative immediately, but they are not adopted by the user program.

Refer to the SILworX communication manual (HI 801 101 E) for more details about configuring safe**ethernet**.

## 7.8 Configuring Alarms and Events

**To define the events**

1. Define a global variable for each event. Generally use global variables that have already been defined for the program.
2. Below the resource, create a new **Alarms&Events** branch, if not existing.
3. Define events in the A&E Editor.
   - Drag global variables onto the event window for Boolean or scalar events.
   - Define the details of the events, see the following tables.

The events are defined.

For further information, refer to the SILworX online help.

The parameters of the Boolean events must be entered in a table with the following columns:

| Column | Description | Range of values |
|---|---|---|
| Name | Name for the event definition; it must be unique within the resource. | Text, max. 32 characters |
| Global Variable | Name of the assigned global variable (added using drag&drop) | |
| Data type | Data type of the global variable; it cannot be modified. | BOOL |
| Event Source | CPU event: The processor module creates the timestamp. It creates all the events in each of its cycles.<br>Auto event: As CPU event<br>Default value: Auto Event | CPU, Auto |
| Alarm when FALSE | Activated: If the global variable value changes from TRUE to FALSE, an event is triggered.<br>Deactivated: If the global variable value changes from FALSE to TRUE, an event is triggered.<br>Default value: Deactivated | Checkbox activated, deactivated |
| Alarm Text | Text specifying the alarm state | Text |
| Alarm priority | Priority of the alarm state<br>Default value: 500 | 0...1000 |
| Alarm Acknowledgment Successful | Activated: The alarm state must be confirmed by the user (acknowledgement)<br>Deactivated: The alarm state may not be confirmed by the user<br>Default value: Deactivated | Checkbox activated, deactivated |
| Return to Normal Text | Text specifying the alarm state | Text |
| Return to Normal Severity | Priority of the normal state<br>Default value: 500 | 0...1000 |
| Return to Normal Ack Required | The normal state must be confirmed by the user (acknowledgement)<br>Default value: Deactivated | Checkbox activated, deactivated |

Table 27: Parameters for Boolean Events

The parameters of the scalar events must be entered in a table with the following columns:

| Column | Description | Range of values |
|---|---|---|
| Name | Name for the event definition; it must be unique within the resource. | Text, max. 32 characters |
| Global Variable | Name of the assigned global variable (added using drag&drop) | |
| Data type | Data type of the global variable; it cannot be modified. | Depending on the global variable type |
| Event Source | CPU event: The processor module creates the timestamp. It creates all the events in each of its cycles.<br>Auto event: As CPU event<br>Default value: Auto Event | CPU, Auto |
| HH Alarm Text | Text specifying the alarm state of the highest limit (HH). | Text |
| HH Alarm Value | Highest limit (HH) triggering an event. Condition:<br>(HH Alarm Value - Hysteresis) > H Alarm Value or HH Alarm Value = H Alarm Value | Depending on the global variable type |
| HH Alarm Priority | Priority of the highest limit (HH); default value: 500 | 0...1000 |
| HH Alarm Acknowledgment Required | Activated: The user must confirm that the highest limit (HH) has been exceeded (acknowledgment).<br>Deactivated: The user may not confirm that the highest limit (HH) has been exceeded.<br>Default value: Deactivated | Checkbox activated, deactivated |
| H Alarm Text | Text specifying the alarm state of the high limit (H). | Text |
| H Alarm Value | High limit (H) triggering an event. Condition:<br>(H Alarm Value - Hysteresis) > (L Alarm Value + Hysteresis) or H Alarm Value = L Alarm Value | Depending on the global variable type |
| H Alarm Priority | Priority of the high limit (H); default value: 500 | 0...1000 |
| H Alarm Acknowledgment Required | Activated: The user must confirm that the high limit (H) has been exceeded (acknowledgment).<br>Deactivated: The user may not confirm that the high limit (H) has been exceeded.<br>Default value: Deactivated | Checkbox activated, deactivated |
| Return to Normal Text | Text specifying the normal state | Text |
| Return to Normal Severity | Priority of the normal state; default value: 500 | 0...1000 |
| Return to Normal Ack Required | The normal state must be confirmed by the user (acknowledgement); default value: Deactivated | Checkbox activated, deactivated |
| L Alarm Text | Text specifying the alarm state of the low limit (L). | Text |
| L Alarm Value | Low limit (L) triggering an event. Condition:<br>(L Alarm Value + Hysteresis) < (H Alarm Value - Hysteresis) or L Alarm Value = H Alarm Value | Depending on the global variable type |
| L Alarm Priority | Priority of low limit (L); default value: 500 | 0...1000 |
| L Alarm Acknowledgment Required | Activated: The user must confirm that the low limit (L) has been exceeded (acknowledgment).<br>Deactivated: The user may not confirm that the low limit (L) has been exceeded.<br>Default value: Deactivated | Checkbox activated, deactivated |
| LL Alarm Text | Text specifying the alarm state of the lowest limit (LL). | Text |
| LL Alarm Value | Lowest limit (LL) triggering an event. Condition:<br>(LL Alarm Value + Hysteresis) < (L Alarm Value) or<br>LL Alarm Value = L Alarm Value | Depending on the global variable type |
| LL Alarm Priority | Priority of the lowest limit (LL); default value: 500 | 0...1000 |

<table>
<tr><th>Column</th><th colspan="2">Description</th><th>Range of values</th></tr>
<tr><td rowspan="3">LL Alarm Acknowledgment Required</td><td>Activated</td><td>The user must confirm that the lowest limit (LL) has been exceeded (acknowledgment).</td><td rowspan="3">Checkbox activated, deactivated</td></tr>
<tr><td>Deactivated</td><td>The user may not confirm that the lowest limit (LL) has been exceeded.</td></tr>
<tr><td colspan="2">Default value: Deactivated</td></tr>
<tr><td>Alarm Hysteresis</td><td colspan="2">The hysteresis avoids that many events are continuously created when the process value often oscillate around a limit.</td><td>Depending on the global variable type</td></tr>
</table>

Table 28: Parameters for Scalar Events

i

**Faulty event recording due to improper parameter settings possible!**

**Setting the parameters *L Alarm Value* and *H Alarm Value* to the same value can cause an unexpected behavior of the event recording since no normal range exists in such a case.**

**For this reason, make sure that *L Alarm Value* and *H Alarm Value* are set to different values.**

## 7.9 Handling the User Program

The PADT can be used to influence the program's function within the controller as follows:

### 7.9.1 Setting the Parameters and the Switches

During the user program's configuration, the parameters and the switches are set to offline and are loaded into the controller with the code-generated program. The parameters and the switches can also be set when the controller is in the STOP or RUN state, provided that the *main enable* switch has been activated. Only the elements in the NVRAM can be modified, all remaining elements are activated during the load procedure.

### 7.9.2 Starting the Program from STOP/VALID CONFIGURATION

Starting the program has the same effect as switching the controller's mode of operation from STOP/VALID CONFIGURATION to RUN; the program enters the RUN state too. The program enters the test mode if the test mode is active while starting the program. In accordance with IEC 61131, a cold or a warm start can also be performed in addition the starting in test mode.

i

The program can only be started if the *Start/Restart Allowed* switch was activated.

### 7.9.3 Restarting the Program after Errors

If the program enters the STOP/INVALID CONFIGURATION state, e.g., due to unauthorized access to operating system areas, it restarts. If the user program enters the STOP/INVALID CONFIGURATION state again within roughly one minute since the restart, it remains in this state. If this is the case, it can be restarted using the Control Panel's start button. After a restart, the operating system checks the entire program.

If automatic restarting is not desired for an application, the system parameter *Autostart* must be set to OFF.

### 7.9.4 Stopping the Program

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

## 7.9.5 Program Test Mode

The test mode is started from within the Control Panel, selecting **Test Mode** -> **Test Mode with Hot Start** (...Cold Start, ...Warm Start). Each Single Cycle command is used to activate a single cycle (one complete logic cycle).

Behavior of variable/signal values in test mode

The selection of cold, warm or hot start determines which variable values are used during the first cycle in test mode.

Cold start: all variables/signals are set to their initial values.

Warm start: retain signals retain their value, the remaining signals are set to their initial value.

Hot start: All variables/signals retain their current values.

Finally, the Cycle Step command can be used to start the user program in single step mode. All current values are retained for the following cycle (freezed state).

**⚠ WARNING**

**Property damage or physical injury possible due to actuators in unsafe state!**

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

## 7.9.6 Online Test

The *Online Test* function is used to add online test fields (OLT fields) to the program logic, to display and to force signals/variables while the controller is operating.

If the *Online Test Allowed* switch is on, the values of signals/variables can be manually entered in the corresponding OLT fields and thus forced. However, the forced values only apply until they are overwritten by the program logic.

If the *Online Test Allowed* switch is off, the values of the signals/variables in OLT fields are only displayed and cannot be modified.

For more information on how to use OLT fields, enter `OLT field` in the online help of the programming tool.

# 8 Operation

This chapter describes how to handle and diagnose the controller during its operation.

## 8.1 Handling

The controller needs not be handled during its normal operation. Only if problems arise, an intervention with the PADT may be required.

## 8.2 Diagnosis

A first, rough diagnosis can be performed via the light-emitting diodes (LEDs). The diagnostic history provides a more detailed analysis of the operating or error state. It can be displayed using the programming tool.

### 8.2.1 Light-Emitting Diode Indicators

The light emitting diodes (LEDs) indicate the module state. Function and meaning of the LEDs depend on the processor system's operating system currently in use. Refer to the module-specific manuals for details.

The function and meaning of the fieldbus LEDs are described in the SILworX communication manual (HI 801 101 E').

### 8.2.2 Diagnostic History

The diagnostic history records the various states of the processor module and communication system and stores them in a non-volatile memory. Both systems include a short term and a long term diagnosis. The number of entries differs for hardware layout and the operating system versions:

| | CPU |
|---|---|
| Number of entries in the long term diagnosis | 700 |
| Number of entries in the short term diagnosis | 700 |

Table 29: Maximum Number of Entries in the Diagnostic History for M-CPU 01

The long-term diagnosis of the processor system includes the following events:

- Reboot
- Changed mode of operation
  (INIT, RUN, STOP/VALID CONFIGURATION, STOP/INVALID CONFIGURATION),
- Changed program mode of operation
  (START, RUN, ERROR, TEST MODE),
- Configuration load or deletion
- Configuration and reset of switches
- Processor system failures
- Operating system download
- Forcing (setting and resetting the force switch is allowed)
- I/O module diagnostics
- Power supply and temperature diagnostics

The long-term diagnosis of the communication system includes the following events:

- Reboot of the communication system
- Changed mode of operation
  (INIT, RUN, STOP/VALID CONFIGURATION, STOP/INVALID CONFIGURATION),
- User log-in
- Operating system load

If the memory for the long term diagnosis is full, all data older than three days is deleted allowing new entries to be stored. If no data is older than three days, the new entries cannot be stored and get lost. A message in the long-term diagnosis warns that it was not possible to store the data.

The short-term diagnosis of the processor system includes the following events:

- Processor system diagnostics (setting the force switches and force values)
- User program diagnostics (cyclic operation)
- Communication diagnostics
- Power supply and temperature diagnostics
- I/O module diagnostics

The short-term diagnosis of the communication system includes the following events:

- safe**ethernet**-related events
- Start / stop while writing to the flash memory
- Faults that can occur while loading a configuration from the flash memory
- Unsuccessful time synchronization between the communication system and the processor system

Parameter errors associated with the inputs or outputs might be not detected during the code generation. If a parameter error occurs, the message INVALID CONFIGURATION with the error source and code are displayed in the feedback box. This message helps analyzing errors due to an incorrect configuration of the inputs or outputs.

If the memory for the short-term diagnosis is full, the oldest entries are deleted to allow new data to be saved. No message appears warning that old entries are being deleted.

Diagnostic data recording is not safety-related. To read the data recorded in chronological order, use the programming tool. Reading does not delete the data stored in the controller. The programming tool is capable of storing the contents of the diagnostic window.

### 8.2.3 Diagnosis in SILworX

The online view in the SILworX Hardware Editor is used to access to the diagnostic panel.

**To open the diagnostic panel**

1. Select the **Hardware** element within the desired resource.
2. Click **Online** on the context menu or on the Action Bar.
   ☑ The system log-in window appears.
3. In the system log-in window, select or enter the following information:
   - IP address of the controller
   - User and password

   ☑ The Hardware Editor's online view appears.
4. In the online view, select the required module, usually the processor or the communication module.
5. Select **Diagnosis** from the context menu or the **Online** menu.

The diagnostic panel for the required module appears.

If a controller is operating, messages about the state of the processor system, communication system and I/O modules are displayed at specific, user-defined time intervals.

# 9 Maintenance

The maintenance of HIMatrix M45 systems is restricted to the following:

- Removing disturbances
- Loading operating systems and user programs

## 9.1 Interferences

Disturbances in the M-CPU 01 processor system mostly result in the complete shut-down of the controller and are indicated via the *ERR* LED on the M-CPU 01 module.

For possible reasons for a lit *ERR* LED, refer to the M-CPU 01 module-specific manual (HI 800 655 E).

To turn off the indicators, start the **Reboot Resource** command located in the **Extra** menu associated with the Control Panel. The controller is booted and re-started.

The system automatically detects disturbances in the input and output channels during operation and displays them via the *FAULT* LED on the front plate of the corresponding module.

Even if the controller is stopped, the PADT diagnostic history can be used to read out detected faults, provided that communication was not disturbed as well.

Prior to replacing a module, check whether an external line disturbance exists and whether the corresponding sensor or actuator is ok.

## 9.2 Loading Operating Systems

The processor and communication modules have different operating systems that are stored in the rewritable flash memories and can be replaced, if necessary.

**NOTE**

**Disruption of the safety-related operation!**

**The controller must be in the STOP state to enable the programming tool to load new operating systems.**

**During this time period, the operator must ensure the plant safety, e.g., by taking organizational measures.**

- The programming tool prevents controllers from loading the operating systems in the RUN state and reports this as such.
- Interruption or incorrect termination of the loading process causes the controller to be no longer functional. It is possible, however, to reload the operating system.

The operating system for the processor system (CPU operating system) must be loaded before that for the communication system (COM operating system).

Requirements for loading operating systems:

- The new operating system must be stored in a directory that can be accessed by the programming tool.
- The PADT is connected to an Ethernet interface of the M-CPU 01 processor module.

**To load the new operating system**

1. Set the controller to the STOP state, if it has not already been done.
2. Open the online view of the hardware and log in to the controller with administrator rights.
3. Right-click the module, processor or communication module.
4. The context menu appears. Click **Maintenance/Service->Load Module Operating System**.

5. In the dialog box *Load Module Operating System,* select the type of the operating system to be loaded.
6. A dialog box for selecting a file appears. Select the file with the operating system that should be loaded and click **Open**.

SILworX loads the new operating system into the controller.

After loading, the loaded operating system version must be verified using the PADT.

## 9.3 Repairing Modules and Sockets

The operator is not authorized to repair modules and sockets of the HIMatrix M45 system. Defective HIMatrix M45 modules must be returned to HIMA for repair after being tested by the operator with a brief description of the fault.

Equipment with a safety certificate is safety-relevant. The validity of the certificate expires if unauthorized repair is performed to safety-related devices of the HIMatrix system.

The warranty is void and no legal responsibility is taken for unauthorized repair.

# 10 Decommissioning

Remove the supply voltage to decommission the modular controller. Afterwards it is possible to pull out the pluggable screw terminal connector blocks for inputs and outputs and the Ethernet cables, and to remove the module.

# 11 Transport

To avoid mechanical damage, HIMatrix components must be transported in packaging.

Always store HIMatrix components in their original product packaging. This packaging also provides protection against electrostatic discharge. Note that the product packaging alone is not suitable for transport.

# 12 Disposal

Industrial customers are responsible for correctly disposing of decommissioned HIMatrix hardware. Upon request, a disposal agreement can be arranged with HIMA.

All materials must be disposed of in an ecologically sound manner.

# Appendix

## Glossary

| Term | Description |
|---|---|
| ARP | Address resolution protocol: Network protocol for assigning the network addresses to hardware addresses |
| AI | Analog input |
| AO | Analog output |
| COM | Communication module |
| CRC | Cyclic redundancy check |
| DI | Digital input |
| DO | Digital output |
| ELOP II Factory | Programming tool for HIMatrix systems |
| EMC | Electromagnetic compatibility |
| EN | European norm |
| ESD | Electrostatic discharge |
| FB | Fieldbus |
| FBD | Function block diagrams |
| FTT | Fault tolerance time |
| ICMP | Internet control message protocol: Network protocol for status or error messages |
| IEC | International electrotechnical commission |
| MAC Address | Media access control address: Hardware address of one network connection |
| PADT | Programming and debugging tool (in accordance with IEC 61131-3),<br>PC with SILworX or ELOP II Factory |
| PE | Protective earth |
| PELV | Protective extra low voltage |
| PES | Programmable electronic system |
| R | Read: The system variable or signal provides value, e.g., to the user program |
| Rack ID | Base plate identification (number) |
| Interference-free | Supposing that two input circuits are connected to the same source (e.g., a transmitter). An input circuit is termed *interference-free* if it does not distort the signals of the other input circuit. |
| R/W | Read/Write (column title for system variable/signal type) |
| SELV | Safety extra low voltage |
| SFF | Safe failure fraction, portion of faults that can be safely controlled |
| SIL | Safety integrity level (in accordance with IEC 61508) |
| SILworX | Programming tool for HIMatrix systems |
| SNTP | Simple network time protocol (RFC 1769) |
| SRS | System.Rack.Slot addressing of a module |
| SW | Software |
| TMO | Timeout |
| W | Write: System variable/signal is provided with value, e.g., from the user program |
| $r_{PP}$ | Peak-to-peak value of a total AC component |
| Watchdog (WD) | Time monitoring for modules or programs. If the watchdog time is exceeded, the module or program enters the STOP_ERROR state. |
| WDT | Watchdog time |

## Index of Figures

## Index of Tables

## Declaration of conformity

For the HIMatrix M45 system, declarations of conformity exist for the following directives:

- EMC Directive
- Low Voltage Directive
- EX Directive

The current declarations of conformity are available on the HIMA website `www.hima.com`.

## Index

HIMA Paul Hildebrandt GmbH
P.O. Box 1261
68777 Brühl, Germany
Phone: +49 6202 709-0
Fax: +49 6202 709-107
E-mail: info@hima.com Internet: www.hima.com

(1404)