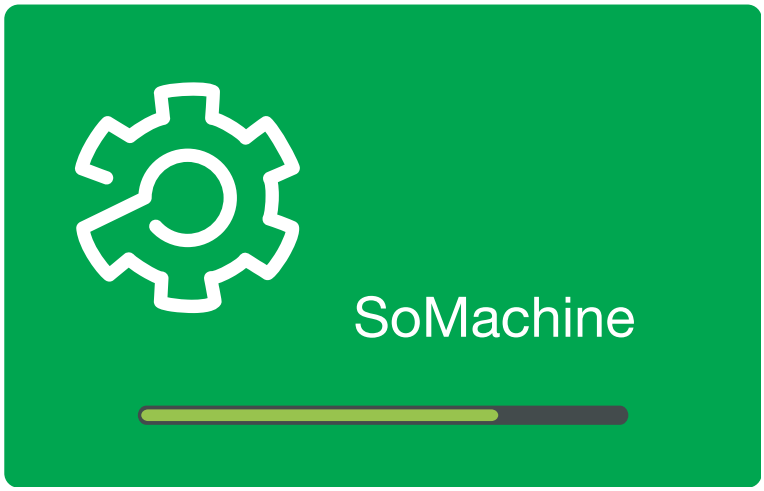
# ILX Library

## Function blocks

## Software manual

V2.09, 04.2012




0198441113886, V2.09, 04.2012

www.schneider-electric.com

Schneider Electric

# Important information

This manual is part of the product.

Carefully read this manual and observe all instructions.

Keep this manual for future reference.

Hand this manual and all other pertinent product documentation over to all users of the product.

Carefully read and observe all safety instructions and the chapter "Before you begin - safety information".

Some products are not available in all countries.
For information on the availability of products, please consult the catalog.

Subject to technical modifications without notice.

All details provided are technical data which do not constitute warranted qualities.

Most of the product designations are registered trademarks of their respective owners, even if this is not explicitly indicated.

0198441113886, V2.09, 04.2012

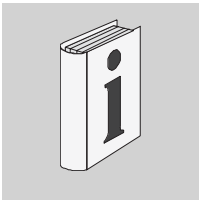
# Table of contents



0198441113886, V2.09, 04.2012

0198441113886, V2.09, 04.2012

# About this manual



This manual is an extract of the SoMachine Online Help. Fully read and understand all manuals of the SoMachine Online Help and of the products used.

*Purpose of this document*

This document describes the functions of the Integrated Lexium Library.

| Software environment | Devices | Fieldbus |
|---|---|---|
| SoMachine<br>Device Descriptions of version 3.0 and higher are supported. | ILA1<br>ILE1<br>ILS1 | CANopen |

*Validity note*

This document is valid for SoMachine as of Version 2.0.

*Source manuals* The latest versions of the manuals can be downloaded from the Internet at:

http://www.schneider-electric.com

*Corrections and suggestions* We always try to further optimize our manuals. We welcome your suggestions and corrections.

Please get in touch with us by e-mail:
techcomm@schneider-electric.com.

*Work steps* If work steps must be performed consecutively, this sequence of steps is represented as follows:

- ■ Special prerequisites for the following work steps
- ▶ Step 1
- ◁ Specific response to this work step
- ▶ Step 2

If a response to a work step is indicated, this allows you to verify that the work step has been performed correctly.

Unless otherwise stated, the individual steps must be performed in the specified sequence.

*SI units* SI units are the original values. Converted units are shown in brackets behind the original value; they may be rounded.

Example:
Minimum conductor cross section: 1.5 $mm^2$ (AWG 14)

*Glossary* Explanations of special technical terms and abbreviations.

*Index* List of keywords with references to the corresponding page numbers.

0198441113886, V2.09, 04.2012

*Disclaimer*

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products described here. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any user or integrator to perform the appropriate and fully comprehensive risk analyses, evaluation and testing of the products with respect to the relevant specific application or use of the products. Neither Schneider Electric nor any of its affiliate or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.



All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

0198441113886, V2.09, 04.2012

# 1 Before you begin - safety information

1

## 1.1 Qualification of personnel

Only appropriately trained persons who are familiar with and understand the contents of this manual and all other pertinent product documentation are authorized to work on and with this product. In addition, these persons must have received safety training to recognize and avoid hazards involved. These persons must have sufficient technical training, knowledge and experience and be able to foresee and detect potential hazards that may be caused by using the product, by changing the settings and by the mechanical, electrical and electronic equipment of the entire system in which the product is used.

All persons working on and with the product must be fully familiar with all applicable standards, directives, and accident prevention regulations when performing such work.

## 1.2 Intended use

This product is a library for industrial use with the appropriate controllers and drives.

The product may only be used in compliance with all applicable safety regulations and directives, the specified requirements and the technical data.

Prior to using the product, you must perform a risk assessment in view of the planned application. Based on the results, the appropriate safety measures must be implemented.

Since the product is used as a component in an entire system, you must ensure the safety of persons by means of the design of this entire system (for example, machine design).

Any use other than the use explicitly permitted is prohibited and can result in hazards.

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel.

0198441113886, V2.09, 04.2012

## 1.3 Hazard categories

Safety instructions to the user are highlighted by safety alert symbols in the manual. In addition, labels with symbols and/or instructions are attached to the product that alert you to potential hazards.

Depending on the seriousness of the hazard, the safety instructions are divided into 4 hazard categories.

**⚠ DANGER**

DANGER indicates an imminently hazardous situation, which, if not avoided, **will result** in death or serious injury.

**⚠ WARNING**

WARNING indicates a potentially hazardous situation, which, if not avoided, **can result** in death, serious injury, or equipment damage.

**⚠ CAUTION**

CAUTION indicates a potentially hazardous situation, which, if not avoided, **can result** in injury or equipment damage.

**CAUTION**

CAUTION used without the safety alert symbol, is used to address practices not related to personal injury (e.g. **can result** in equipment damage).

0198441113886, V2.09, 04.2012

## 1.4 Basic information

**⚠ WARNING**

**LOSS OF CONTROL**

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop, overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical functions.
- System control paths may include communication links. Consideration must be given to the implication of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines. [1)]
- Each implementation of the product must be individually and thoroughly tested for proper operation before being placed into service.

**Failure to follow these instructions can result in death or serious injury.**

1) For USA: Additional information, refer to NEMA ICS 1.1 (latest edition), “Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control” and to NEMA ICS 7.1 (latest edition), “Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems”.

**⚠ WARNING**

**UNINTENDED BEHAVIOR DUE TO IMPROPER ERROR HANDLING**

Improper error handling can change movements or signals or deactivate monitoring functions.

- Carefully program the error handling routines.
- Verify the effectiveness of error handling.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

**⚠ WARNING**

**UNINTENDED BEHAVIOR DUE TO CHANGES TO THE LIBRARY**

- Do not change or manipulate the library in any way whatsoever.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

0198441113886, V2.09, 04.2012

## 1.5 Standards and terminology

Technical terms, terminology and the corresponding descriptions in this manual are intended to use the terms or definitions of the pertinent standards.

In the area of drive systems, this includes, but is not limited to, terms such as "safety function", "safe state", "fault", "fault reset", "failure", "error", "error message", "warning", "warning message", etc.

Among others, these standards include:

- IEC 61800: "Adjustable speed electrical power drive systems"
- IEC 61158: "Digital data communications for measurement and control – Fieldbus for use in industrial control systems"
- IEC 61784: "Industrial communication networks – Profiles"
- IEC 61508: "Functional safety of electrical/electronic/programmable electronic safety-related systems"

Also see the glossary at the end of this manual.

0198441113886, V2.09, 04.2012

# 2 Integrated Lexium Library Guide

2

Edition: V1.02

*Library name* Integrated Lexium Library (ILX)

| Software environment | Devices | Fieldbus |
|---|---|---|
| SoMachine<br>Device Descriptions of version 3.0 and higher are supported. | ILA1<br>ILE1<br>ILS1 | CANopen |

The function blocks described here are used to control Lexium Integrated Drives ILX in CANopen fieldbuses under the SoMachine software environment. The function blocks are compliant with the IEC 61131-3 standard.

*Naming conventions*

- Function blocks with the prefix MC_ ("Motion Control") are compliant with the PLCopen specifications. They conform to a global standard for programming motion control applications.
- Function blocks without a prefix are vendor-specific (Schneider Electric); however, they comply with the general PLC open rules.

*Simple application*

- The function blocks are used in the same way.
- The function blocks comply with the PLCopen state diagram.
- The function blocks feature a visualization that can be easily integrated into the application.

*Categorization of the function blocks*

- Single axis: These function blocks are used for movements or functions of a single, independent axis.
- Multi axis: These function blocks are used for coordinated movements of several axes (for example, Electronic Gear).
- Administrative: These function blocks are used for configuration tasks (such as reading and writing of parameters, restoring a device configuration, etc.).

0198441113886, V2.09, 04.2012

## 2.1 Before you begin - safety information

This page provides safety information that you must be familiar with before you may work with the product.

| |
|---|
| "Qualification of personnel" |
| "Intended use" |
| "Hazard categories" |
| "Basic information" |
| "Standards and terminology" |

*Qualification of personnel*

Only appropriately trained persons who are familiar with and understand the contents of this manual and all other pertinent product documentation are authorized to work on and with this product. In addition, these persons must have received safety training to recognize and avoid hazards involved. These persons must have sufficient technical training, knowledge and experience and be able to foresee and detect potential hazards that may be caused by using the product, by changing the settings and by the mechanical, electrical and electronic equipment of the entire system in which the product is used.

All persons working on and with the product must be fully familiar with all applicable standards, directives, and accident prevention regulations when performing such work.

*Intended use*

This product is a library for industrial use with the appropriate controllers and drives.

The product may only be used in compliance with all applicable safety regulations and directives, the specified requirements and the technical data.

Prior to using the product, you must perform a risk assessment in view of the planned application. Based on the results, the appropriate safety measures must be implemented.

Since the product is used as a component in an entire system, you must ensure the safety of persons by means of the design of this entire system (for example, machine design).

Any use other than the use explicitly permitted is prohibited and can result in hazards.

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel.

*Hazard categories*

Safety instructions to the user are highlighted by safety alert symbols in the manual. In addition, labels with symbols and/or instructions are attached to the product that alert you to potential hazards.

Depending on the seriousness of the hazard, the safety instructions are divided into 4 hazard categories.

0198441113886, V2.09, 04.2012

**⚠ DANGER**

DANGER indicates an imminently hazardous situation, which, if not avoided, **will result** in death or serious injury.

**⚠ WARNING**

WARNING indicates a potentially hazardous situation, which, if not avoided, **can result** in death, serious injury, or equipment damage.

**⚠ CAUTION**

CAUTION indicates a potentially hazardous situation, which, if not avoided, **can result** in injury or equipment damage.

**CAUTION**

CAUTION used without the safety alert symbol, is used to address practices not related to personal injury (e.g. **can result** in equipment damage).

*Basic information*

**⚠ WARNING**

**LOSS OF CONTROL**

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop, overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical functions.
- System control paths may include communication links. Consideration must be given to the implication of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines. [1)]
- Each implementation of the product must be individually and thoroughly tested for proper operation before being placed into service.

**Failure to follow these instructions can result in death or serious injury.**

1) For USA: Additional information, refer to NEMA ICS 1.1 (latest edition), “Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control” and to NEMA ICS 7.1 (latest edition), “Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems”.

0198441113886, V2.09, 04.2012

**⚠ WARNING**

**UNINTENDED BEHAVIOR DUE TO IMPROPER ERROR HANDLING**

Improper error handling can change movements or signals or deactivate monitoring functions.

- Carefully program the error handling routines.
- Verify the effectiveness of error handling.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

**⚠ WARNING**

**UNINTENDED BEHAVIOR DUE TO CHANGES TO THE LIBRARY**

- Do not change or manipulate the library in any way whatsoever.

**Failure to follow these instructions can result in death, serious injury or equipment damage.**

*Standards and terminology*

Technical terms, terminology and the corresponding descriptions in this manual are intended to use the terms or definitions of the pertinent standards.

In the area of drive systems, this includes, but is not limited to, terms such as "safety function", "safe state", "fault", "fault reset", "failure", "error", "error message", "warning", "warning message", etc.

Among others, these standards include:

- IEC 61800: "Adjustable speed electrical power drive systems"
- IEC 61158: "Digital data communications for measurement and control – Fieldbus for use in industrial control systems"
- IEC 61784: "Industrial communication networks – Profiles"
- IEC 61508: "Functional safety of electrical/electronic/programmable electronic safety-related systems"

Also see the glossary at the end of this manual.

## 2.2 About the book

### 2.2.1 Purpose of this document

This document describes the functions of the Integrated Lexium Library.

| Software environment | Devices | Fieldbus |
|---|---|---|
| SoMachine<br>Device Descriptions of version 3.0 and higher are supported. | ILA1<br>ILE1<br>ILS1 | CANopen |

0198441113886, V2.09, 04.2012

### 2.2.2 Validity note

This document is valid for SoMachine as of Version 2.0.

### 2.2.3 Disclaimer

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products described here. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any user or integrator to perform the appropriate and fully comprehensive risk analyses, evaluation and testing of the products with respect to the relevant specific application or use of the products. Neither Schneider Electric nor any of its affiliate or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.



All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

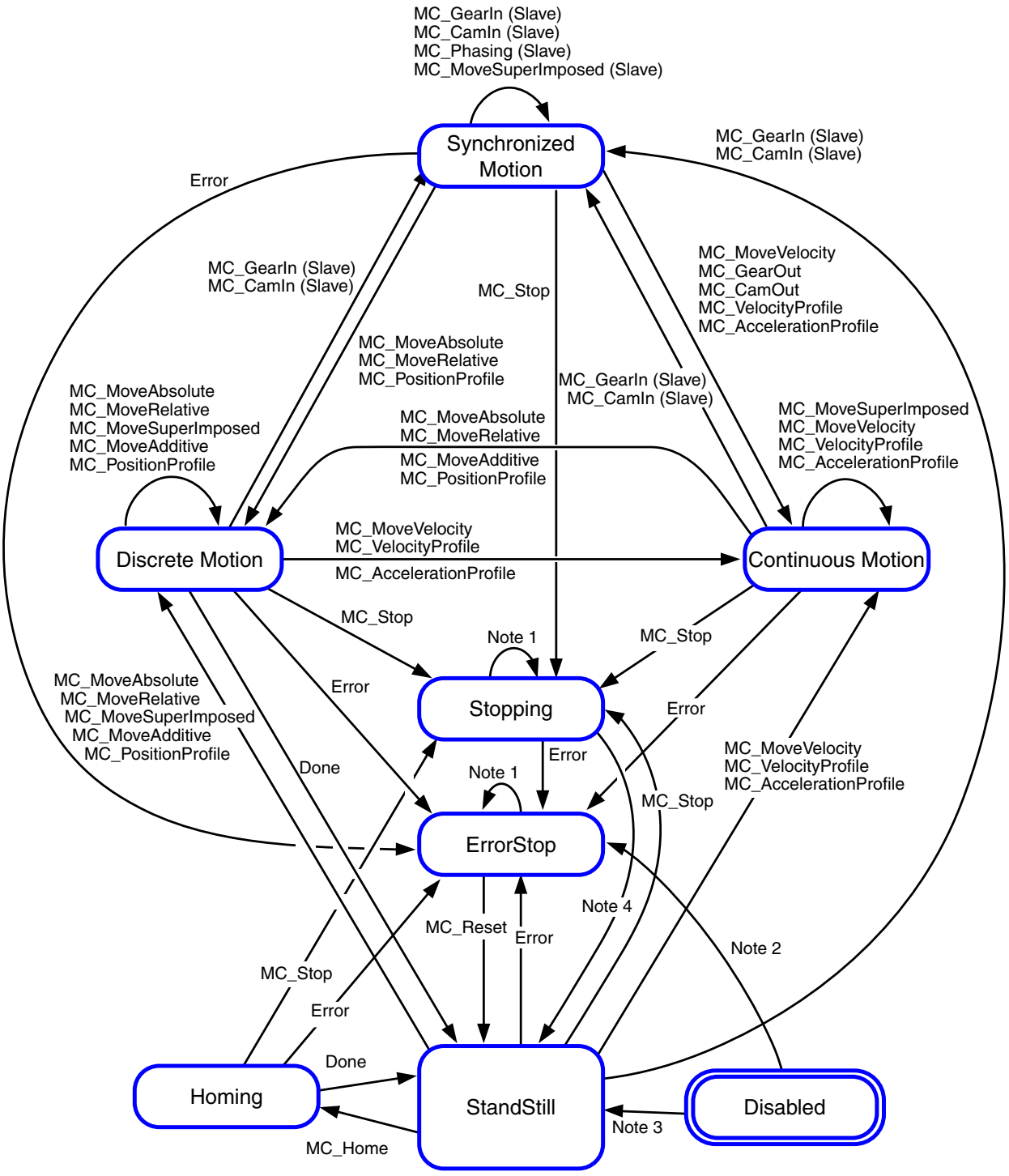
Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

0198441113886, V2.09, 04.2012

## 2.3 PLCopen state diagram

At any given point in time, the drive is exactly in one state. If a function block is executed or an error occurs, this may cause a state transition. The function block "2.8.1.3 MC_ReadStatus_ILX" is used to read the current status of the drive.




Note 1: In the states ErrorStop or Stopping, all function blocks can be called, but none of them will be executed, except for

0198441113886, V2.09, 04.2012

"2.8.5.2 MC_Reset_ILX" and Error. Calling the function block "2.8.5.2 MC_Reset_ILX" will cause a transition to the states StandStill or ErrorStop, respectively.

Note 2: Power.Enable = TRUE and there is an error

Note 3: Power.Enable = TRUE and there is no error

Note 4: "2.6.6.1 MC_Stop_ILX". Done AND NOT "2.6.6.1 MC_Stop_ILX" Execute.

## 2.4 List of the function blocks

| Category | Subcategory | Function block | Type | ILA1 | ILE1 | ILS1 |
|---|---|---|---|---|---|---|
| Single axis | | | | | | |
| | Initialization | "2.6.1.1 MC_Power_ILX" | PLCopen | X | X | X |
| | Operating mode Jog | "2.6.2.1 MC_Jog_ILX" | PLCopen | X | X | X |
| | Operating mode Profile Velocity | "2.6.3.1 MC_MoveVelocity_ILX" | PLCopen | X | X | X |
| | Operating mode Profile Position | "2.6.4.1 MC_MoveAbsolute_ILX" | PLCopen | X | X | X |
| | | "2.6.4.2 MC_MoveAdditive_ILX" | PLCopen | X | X | X |
| | Operating mode Homing | "2.6.5.1 MC_Home_ILX" | PLCopen | X | X | X |
| | | "2.6.5.2 MC_SetPosition_ILX" | PLCopen | X | X | X |
| | Stopping | "2.6.6.1 MC_Stop_ILX" | PLCopen | X | X | X |
| | Position capture via signal input | "2.6.7.1 MC_TouchProbe_ILX" | PLCopen | X | - | X |
| | | "2.6.7.2 MC_AbortTrigger_ILX" | PLCopen | X | - | X |

| Category | Subcategory | Function block | Type | ILA1 | ILE1 | ILS1 |
|---|---|---|---|---|---|---|
| Multi axis | | | | | | |
| | Operating mode Electronic Gear | "2.7.1.1 GearInSync_ILX" | Vendor-specific | X | - | - |
| | | "2.7.1.2 MC_GearOut_ILX" | PLCopen | X | - | - |

0198441113886, V2.09, 04.2012

| Category | Subcategory | Function block | Type | ILA1 | ILE1 | ILS1 |
|---|---|---|---|---|---|---|
| Administrative | | | | | | |
| | Reading a parameter | "2.8.1.1 MC_ReadActualVelocity_ILX" | PLCopen | X | X | X |
| | | "2.8.1.2 MC_ReadActualPosition_ILX" | PLCopen | X | X | X |
| | | "2.8.1.3 MC_ReadStatus_ILX" | PLCopen | X | X | X |
| | | "2.8.1.4 MC_ReadParameter_ILX" | PLCopen | X | X | X |
| | | "2.8.1.5 GetSupplierVersion" | Vendor-specific | X | X | X |
| | Writing a parameter | "2.8.2.1 MC_WriteParameter_ILX" | PLCopen | X | X | X |
| | | "2.8.2.2 SetDriveRamp_ILX" | Vendor-specific | X | X | X |
| | | "2.8.2.3 ResetParameters_ILX" | Vendor-specific | X | X | X |
| | | "2.8.2.4 StoreParameters_ILX" | Vendor-specific | X | X | X |
| | Saving and restoring device configuration | "2.8.3.1 UploadDriveParameter_ILX" | Vendor-specific | X | X | X |
| | | "2.8.3.2 DownloadDriveParameter_ILX" | Vendor-specific | X | X | X |
| | Inputs and outputs | "2.8.4.1 ConfigureIO_ILX" | Vendor-specific | X | X | X |
| | | "2.8.4.2 ControlIO_ILX" | Vendor-specific | X | X | X |
| | | "2.8.4.3 MC_ReadDigitalInput_ILX" | PLCopen | X | X | X |
| | | "2.8.4.4 MC_ReadDigitalOutput_ILX" | PLCopen | X | X | X |
| | | "2.8.4.5 MC_WriteDigitalOutput_ILX" | PLCopen | X | X | X |
| | Error handling | "2.8.5.1 MC_ReadAxisError_ILX" | PLCopen | X | X | X |
| | | "2.8.5.2 MC_Reset_ILX" | PLCopen | X | X | X |

0198441113886, V2.09, 04.2012

## 2.5 Basic inputs and outputs

| Input/output | Data type | Description |
|---|---|---|
| Axis | Axis_Ref_ILX | Name of the axis (instance) for which the function block is to be executed. The name must be declared in the PLC configuration. The name of the axis can be found to the left in the tree structure of your software. |
| Input | Input_Ref_ILX | Input is a special data type for digital and analog inputs. The data type corresponds to the name of the axis (instance) to which the inputs belong (similar to Axis).<br>In the case of function blocks specifically provided for reading analog and digital inputs, Input replaces the input Axis. |
| Output | Output_Ref_ILX | Output is a special data type for digital and outputs. The data type corresponds to the name of the axis (instance) to which the outputs belong (similar to Axis).<br>In the case of function blocks specifically provided for writing and reading analog and digital inputs, Output replaces the input Axis. |

| Input | Data type | Description |
|---|---|---|
| Enable | BOOL | Value range: TRUE, FALSE<br>Initial value: FALSE<br>The input Enable starts or terminates the execution of a function block. (exception "2.6.1.1 MC_Power_ILX")<br>FALSE: Execution of the function block is terminated. The outputs Valid, Busy, CommandAborted and Error are set to FALSE.<br>TRUE: The function block is executed repeatedly. |
| Execute | BOOL | Value range: TRUE, FALSE<br>Initial value: FALSE<br>The input Execute starts the execution of a function block in the case of a rising edge (FALSE->TRUE).<br>If a second rising edge is detected during the execution of the function block, the current execution is aborted and the function block is executed again.<br>Execution is terminated as soon as the output Busy is FALSE.<br>FALSE and, at the same time, Busy = FALSE:<br>Either Done, Error or CommandAborted are set to TRUE for one call.<br>TRUE and, at the same time, Busy = FALSE:<br>Either Done, Error or CommandAborted are set to TRUE and remain TRUE until Execute is set to FALSE. |

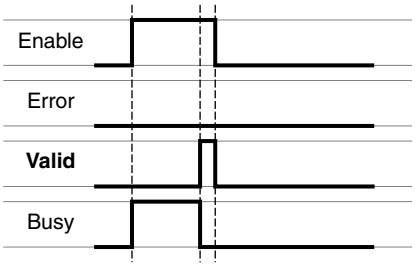
0198441113886, V2.09, 04.2012

| Output | Data type | Description |
|---|---|---|
| Done | BOOL | Value range: TRUE, FALSE<br>Initial value: FALSE<br>FALSE: Execution has not (yet) been terminated without an error.<br>TRUE: Execution has been completed without an error. |
| Valid | BOOL | Value range: TRUE, FALSE<br>Initial value: FALSE<br>FALSE: Execution has not (yet) been terminated without an error. The values at the outputs are not (yet) valid.<br>TRUE: Execution has been completed without an error. The values at the outputs are valid and can be further processed. |
| Busy | BOOL | Value range: TRUE, FALSE<br>Initial value: FALSE<br>FALSE: Execution of the function block has been terminated.<br>TRUE: Function block is being executed.<br>NOTE: In the operating mode Profile Velocity, the output remains TRUE even when the target velocity has been reached or Execute becomes FALSE. The output Busy is set to FALSE as soon as another function block such as MC_Stop is executed. |
| CommandAborted | BOOL | Value range: TRUE, FALSE<br>Initial value: FALSE<br>FALSE: Execution has not (yet) been canceled without an error.<br>TRUE: Execution has been aborted by another function block. |
| Error | BOOL | Value range: TRUE, FALSE<br>Initial value: FALSE<br>FALSE: Execution of the function block is running, nor error has occurred up until now.<br>TRUE: An error has occurred in the execution of the function block. |

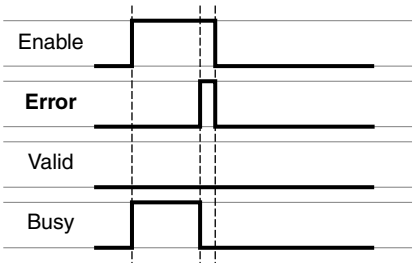
0198441113886, V2.09, 04.2012

### 2.5.1 Signal behavior of function blocks with the input `Enable`

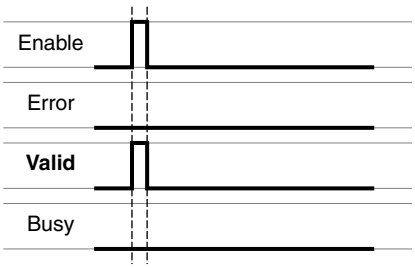
*Example 1* Single execution without error (execution requires more than one call).



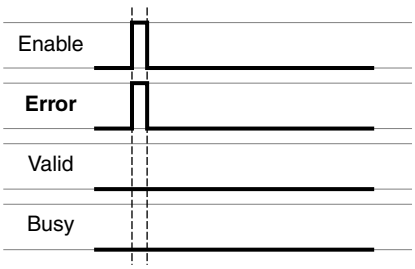

*Example 2* Single execution with error (execution requires more than one call).




*Example 3* Single execution without error (execution requires only one call).

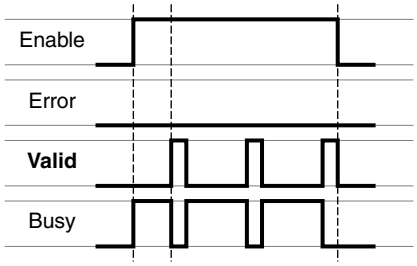



*Example 4* Single execution with error (execution requires only one call).

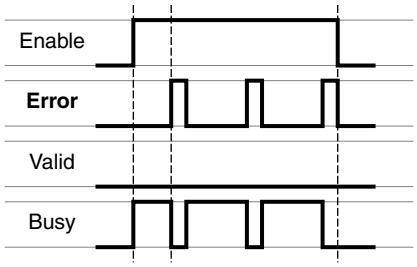



0198441113886, V2.09, 04.2012

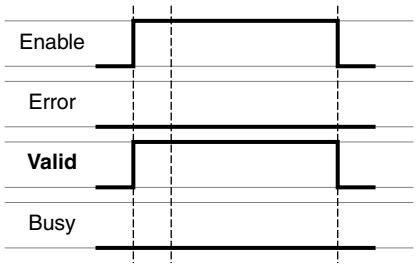
*Example 5* Repeated execution without error (execution requires more than one call).




*Example 6* Repeated execution with error (execution requires more than one call).




*Example 7* Repeated execution without error (execution requires only one call).

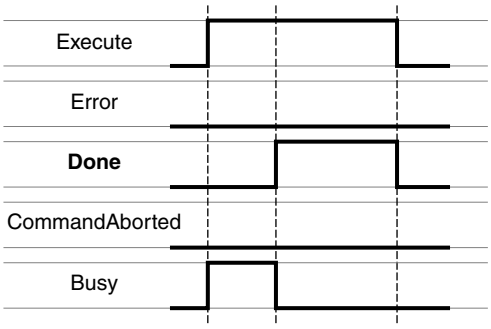



*Example 8* Repeated execution with error (execution requires only one call).

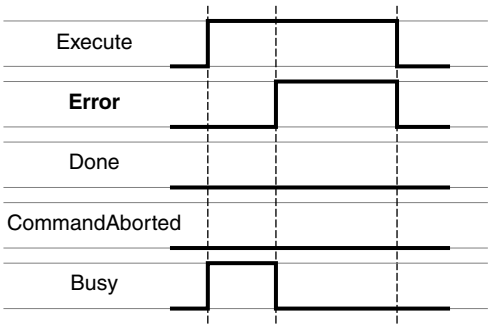

0198441113886, V2.09, 04.2012

### 2.5.2 Signal behavior of function blocks with the input `Execute`

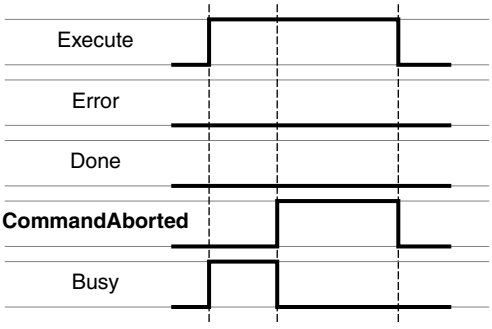
*Example 1* Execution terminated without error.




*Example 2* Execution terminated with error.

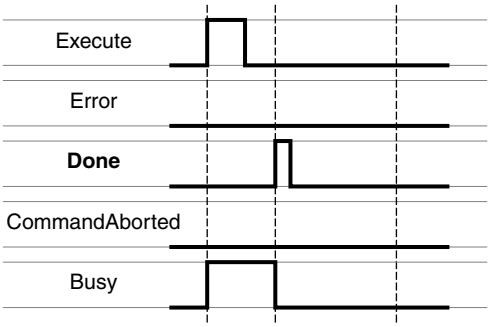



*Example 3* Abortion of the execution because another function block takes over control.




0198441113886, V2.09, 04.2012

*Example 4* Execution completed without error after Execute has been set to FALSE during execution.




0198441113886, V2.09, 04.2012

# 2.6 Single axis

## 2.6.1 Initialization

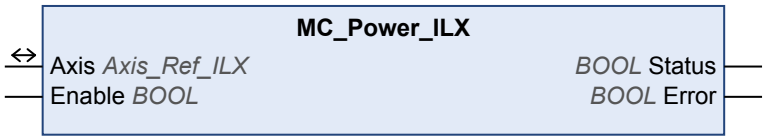
The initialization function block enables or disables the power stage. Other function blocks can only be used when the power stage is enabled.

### 2.6.1.1 MC_Power_ILX

*Function description*

The function block enables or disables the power stage. TRUE at the input `Enable` enables the power stage. Once the power stage is enabled, the output `Status` is set. FALSE at the input `Enable` disables the power stage. Once the power stage is disabled, the output `Status` is reset. If errors occur during execution, the output `Error` is set.

*Graphical representation*




*Compatible devices*

ILA, ILE and ILS

*Inputs/outputs*

The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| `Status` | `BOOL` | Value range: FALSE, TRUE<br>Initial value: FALSE<br>FALSE: Power stage is disabled.<br>TRUE: Power stage is enabled. |

"2.5 Basic inputs and outputs"

*Notes*

In the case of a Node Guarding error, the error memory must be reset by means of the function block "2.8.5.2 MC_Reset_ILX" before the power stage can be enabled again.

If the input Enable = TRUE, an error is signaled if the power supply is lost.

The output `Status` is set to FALSE and the output `Error` to TRUE. Once the power supply is available again, the output `Status` is set back to TRUE.

*Additional information*

"2.3 PLCopen state diagram"

"2.6.1 Initialization"

## 2.6.2 Operating mode Jog

In the operating mode Jog, a movement is made from the actual motor position in the desired direction. The velocity can be set. As long as the signal for the direction is available, a continuous movement is made in the desired direction.

0198441113886, V2.09, 04.2012

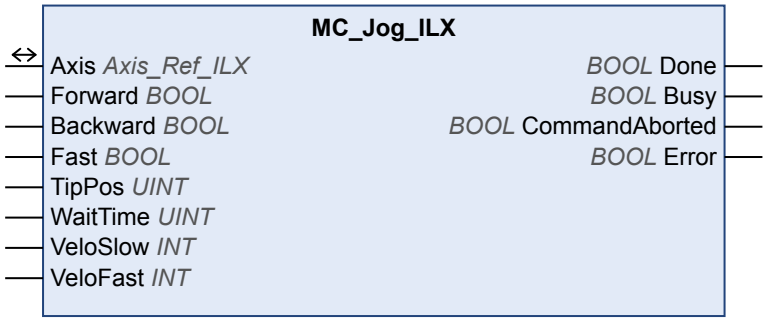
If movements in positive and negative directions are requested at the same time, there is no motor movement.

### 2.6.2.1 MC_Jog_ILX

*Function description*

The function block starts the operating mode Jog. TRUE at the input `Forward` or the input `Backward` starts the jog movement. If both the inputs `Forward` and `Backward` are FALSE, the operating mode is terminated and the output `Done` is set. If both the inputs `Forward` and `Backward` are TRUE, the operating mode remains active, the jog movement is stopped and the output `Busy` remains set.

*Graphical representation*




*Compatible devices*

ILA, ILE and ILS

0198441113886, V2.09, 04.2012

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| `Forward` | `BOOL` | Value range: FALSE, TRUE<br>Initial value: FALSE<br>FALSE: No movement in positive direction<br>TRUE: Movement in positive direction is started. |
| `Backward` | `BOOL` | Value range: FALSE, TRUE<br>Initial value: FALSE<br>FALSE: No movement in negative direction<br>TRUE: Movement in negative direction is started. |
| `Fast` | `BOOL` | Value range: FALSE, TRUE<br>Initial value: FALSE<br>The velocity can be changed during the movement.<br>FALSE: Movement at the velocity set in `VeloSlow`.<br>TRUE: Movement at the velocity set in `VeloFast`. |
| `TipPos` | `UINT` | Value range: 0 ... 65535<br>Initial value: 20<br>0: Continuous movement is started immediately.<br>>0: Movement by this distance value in increments [inc]. The movement is stopped, the waiting time `WaitTime` starts. After the waiting time `WaitTime` has elapsed, a continuous movement is started. |
| `WaitTime` | `UINT` | Value range: 1 ... 10000<br>Initial value: 500<br>Waiting time in [ms]. If `TipPos` is >0, the waiting time `WaitTime` starts as soon as the adjusted distance has been covered. After the waiting time `WaitTime` has elapsed, a continuous movement is started. |
| `VeloSlow` | `INT` | Value range: 1 ... 3000<br>Initial value: 300<br>Velocity in [$min^{-1}$]. If `Fast` = FALSE, the movement is made at this velocity. |
| `VeloFast` | `INT` | Value range: 1 ... 3000<br>Initial value: 1000<br>Velocity in [$min^{-1}$]. If `Fast` = TRUE, the movement is made at this velocity. |

"2.5 Basic inputs and outputs"

*Additional information* "2.3 PLCopen state diagram"

"2.6.2 Operating mode Jog"

## 2.6.3 Operating mode Profile Velocity

You can set a target velocity in the operating mode Profile Velocity. The movement is performed with this target velocity in the operating mode Profile Velocity. The movement continues until a new target velocity is set or until the operating mode is aborted.

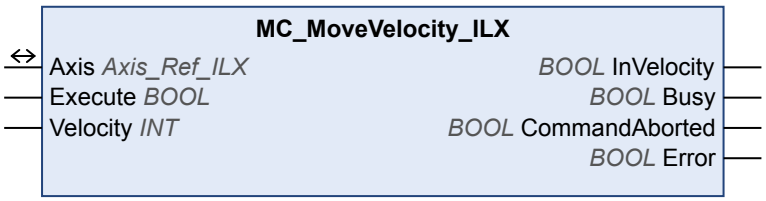
Transitions between two target velocities are performed on the basis of a motion profile. The motion profile is determined by the profile generator in the drive on the basis of the actual velocity, the target velocity and the acceleration and deceleration ramps.

0198441113886, V2.09, 04.2012

### 2.6.3.1 MC_MoveVelocity_ILX

*Function description* The function block starts the operating mode Profile Velocity with the velocity `Velocity`. When the target velocity is reached, `InVelocity` is set.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| `Velocity` | `INT` | Value range:<br>Initial value: 0<br>Unit: [$min^{-1}$]<br>Target velocity |

The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| `InVelocity` | `BOOL` | Value range: FALSE, TRUE<br>Initial value: FALSE<br>FALSE: Target velocity not yet reached.<br>TRUE: Target velocity reached. |

"2.5 Basic inputs and outputs"

*Notes* In the operating mode Profile Velocity, a position overtravel does not trigger an error. A position overtravel results in a loss of the zero point.

*Additional information* "2.3 PLCopen state diagram"

"2.6.3 Operating mode Profile Velocity"

## 2.6.4 Operating mode Profile Position

The following settings can be made in the operating mode Profile Position:

- Target position
- Type of movement (relative movement or absolute movement)
- Target velocity
- Acceleration and deceleration ramps

The movement to the target position is made on the basis of a motion profile. The motion profile is calculated by the profile generator in the drive. The calculation is performed on the basis of the actual position and the target position, the actual velocity and the target velocity and the acceleration and deceleration ramps.

0198441113886, V2.09, 04.2012

In the operating mode Profile Position, absolute movements, relative movements and additive movements are possible.

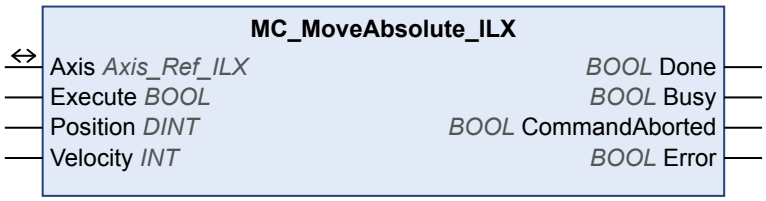
- Absolute movement with reference to the zero point
- Relative movement with reference to the actual position
- Additive movement with reference to the previous target position

A zero point must be defined with the operating mode Homing prior to the first absolute movement.

### 2.6.4.1 MC_MoveAbsolute_ILX

*Function description* The function block starts a movement to the absolute target position `Position` at velocity `Velocity`.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| `Position` | `DINT` | Value range: -2147483648 ... 2147483647<br>Initial value: 0<br><br>Target position absolute in increments.<br>ILA: 16384 increments per revolution<br>ILE: 12 increments per revolution<br>ILS: 20000 increments per revolution |
| `Velocity` | `INT` | Value range:<br>Initial value: 1000<br>Unit: [$min^{-1}$]<br><br>0: Velocity is decelerated until the motor is at a standstill. The function block remains active, the output `Busy` remains TRUE<br>>0: Target velocity value |

"2.5 Basic inputs and outputs"

*Notes*

- Absolute positioning requires a valid zero point. You can use the function block "2.8.1.3 MC_ReadStatus_ILX" to check for a valid zero point.

*Additional information* "2.3 PLCopen state diagram"

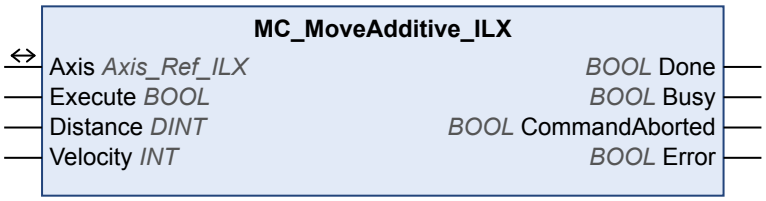
"2.6.4 Operating mode Profile Position"

### 2.6.4.2 MC_MoveAdditive_ILX

*Function description* The function block starts a movement to the original target position plus distance `Distance` at velocity `Velocity`.

0198441113886, V2.09, 04.2012

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| Distance | DINT | Value range: -2147483648 ... 2147483647<br>Initial value: 0<br>Target position relative with reference to the previous target position |
| Velocity | INT | Value range:<br>Initial value: 1000<br>Unit: [$min^{-1}$]<br>Target velocity |

"2.5 Basic inputs and outputs"

*Additional information* "2.3 PLCopen state diagram"

"2.6.4 Operating mode Profile Position"

## 2.6.5 Operating mode Homing

The operating mode Homing is used to define a reference point. The reference point establishes an absolute position reference between the motor position and a defined axis position. The reference point can be defined by means of a reference movement or by means of position setting.

- Reference movement: Movement to a limit switch, a reference switch or the index pulse of the motor encoder. When the position is reached, a position reference is automatically created. This position becomes the absolute user-defined position.
- Position setting: The current motor position is set to a desired position value. The zero point is defined by the position value. Position setting is only possible when the motor is at a standstill.

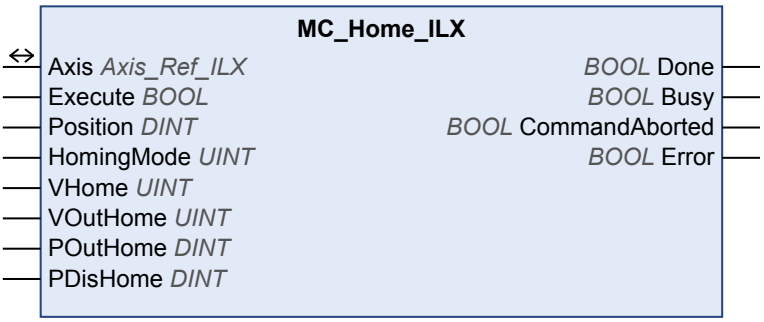
The operating mode Homing must be completed without an error for the new reference point to be valid.

### 2.6.5.1 MC_Home_ILX

*Function description* The function block configures and starts a reference movement.

0198441113886, V2.09, 04.2012

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs.

0198441113886, V2.09, 04.2012

| Input | Data type | Description |
|---|---|---|
| Position | DINT | Value range: -2147483648 ... 2147483647<br>Initial value: 0<br>Value in increments.<br>ILA: 16384 increments per revolution<br>ILE: 12 increments per revolution<br>ILS: 20000 increments per revolution<br>HomingMode 1 ... 8: Position at reference point<br>HomingMode 9: Position for Position Setting |
| HomingMode | UINT | Value range:<br>Initial value: 1<br>1: Positive limit switch (LIMP)<br>2: Negative limit switch (LIMN)<br>3: Reference switch (REF) in negative direction<br>4: Reference switch (REF) in positive direction<br>5: Index pulse in negative direction<br>6: Index pulse in positive direction<br>7: Mechanical stop in negative direction (ILE)<br>8: Mechanical stop in positive direction (ILE) |
| VHome | UINT | Value range:<br>Initial value: 1000<br>Target velocity for searching the switch.<br>HomingMode 1 ... 8 only. |
| VOutHome | UINT | Value range:<br>Initial value: 500<br>Target velocity for moving away from switch. |
| POutHome | DINT | Value range: 1 ... 2147483647<br>Initial value: 200000<br>Maximum distance for search for switching point.<br>0: Search distance monitoring disabled<br>>0: Maximum distance<br>After detection of the switch, the drive starts to search for the defined switching point. If the defined switching point is not found within the distance defined here, the reference movement is canceled with an error. |
| PDisHome | DINT | Value range: 1 ... 2147483647<br>Initial value: 200<br>Maximum search distance after overtravel of switch.<br>0: Search distance monitoring disabled<br>>0: Search distance<br>The switch must be activated again within this search distance, otherwise the reference movement is cancelled. |

"2.5 Basic inputs and outputs"

*Additional information* "2.3 PLCopen state diagram"

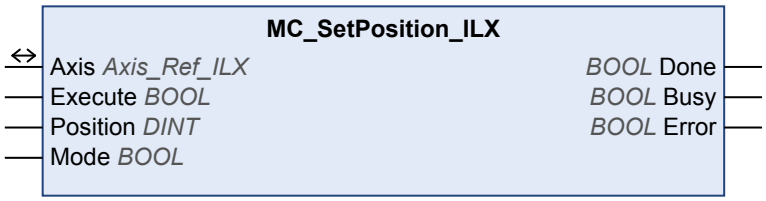
"2.6.5 Operating mode Homing"

### 2.6.5.2 MC_SetPosition_ILX

*Function description* This function block sets a position value at the actual position of the motor. The zero point is defined by the position value. The function block can only be used when the motor is at a standstill.

0198441113886, V2.09, 04.2012

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| `Position` | `DINT` | Value range: -2147483648 ... 2147483647<br>Initial value: 0<br>Value for position setting in increments.<br>ILA: 16384 increments per revolution<br>ILE: 12 increments per revolution<br>ILS: 20000 increments per revolution |
| `Mode` | `BOOL` | Value range: FALSE, TRUE<br>Initial value:<br>FALSE: The actual position is set to the value of the input `Position`.<br>TRUE: The value of the input `Position` is added to the actual position. |

"2.5 Basic inputs and outputs"

*Additional information* "2.6.5 Operating mode Homing"

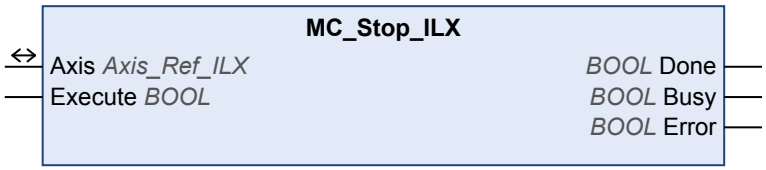
## 2.6.6 Stopping

Each operating mode can be canceled by stopping. Stopping the operating mode does not generate an error.

### 2.6.6.1 MC_Stop_ILX

*Function description* The function block is used to stop the current movement. The operating mode is stopped by the function block.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* "2.5 Basic inputs and outputs"

0198441113886, V2.09, 04.2012

*Notes*

- The deceleration ramp is set with the function block "2.8.2.2 SetDriveRamp_ILX".
- The function block can only be interrupted by disabling the power stage via the function block "2.6.1.1 MC_Power_ILX".
- As long as the input `Execute` is TRUE, no other function block with the exception of "2.6.1.1 MC_Power_ILX" can be started.

*Additional information*

"2.3 PLCopen state diagram"

"2.6.6 Stopping"

## 2.6.7 Position capture via signal input

Position capture via a signal input captures the current position at the point in time at which an edge is detected at one of the digital Capture inputs.

Settings:

- Position capture can be triggered by a rising edge or a falling edge at the signal input.
- It is possible to use one-time or continuous position capture.

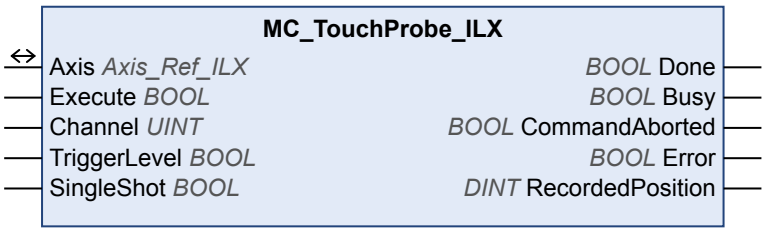
NOTE: This function is not available for Lexium ILE.

### 2.6.7.1 MC_TouchProbe_ILX

*Function description*

The function block configures and starts position capture.

*Graphical representation*




*Compatible devices*

ILA and ILS

*Inputs/outputs*

The table below shows the inputs.

0198441113886, V2.09, 04.2012

| Input | Data type | Description |
|---|---|---|
| Channel | UINT | Value range: 1 ... 2<br>Initial value: 1<br>1: Start position capture via Capture input 1.<br>2: Start position capture via Capture input 2. |
| TriggerLevel | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>FALSE: Start position capture at falling edge.<br>TRUE: Start position capture at rising edge. |
| SingleShot | BOOL | Value range: FALSE, TRUE<br>Initial value: TRUE<br>FALSE: Set continuous position capture. Continuous capture means that the motor position is captured anew at every edge. The previously captured value is lost.<br>TRUE: Sets one-time position capture. One-time capture means that the position is captured at the first edge. The capture value is not overwritten by a new edge. |

The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| RecordedPosition | DINT | Value range: -2147483648 ... 2147483647<br>Initial value: 0<br>Captured motor position |

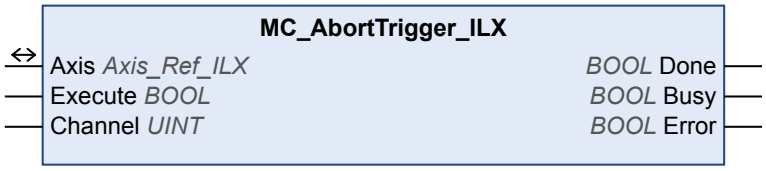
"2.5 Basic inputs and outputs"

*Additional information* "2.6.7 Position capture via signal input"

### 2.6.7.2 MC_AbortTrigger_ILX

*Function description* The function block is used to terminate position capture.

*Graphical representation*




*Compatible devices* ILA and ILS

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| Channel | UINT | Value range: 1 ... 2<br>Initial value: 1<br>1: Cancel position capture via Capture input 1 (IO2, CAP1).<br>2: Cancel position capture via Capture input 2 (IO3, CAP2). |

"2.5 Basic inputs and outputs"

*Additional information* "2.6.7 Position capture via signal input"

0198441113886, V2.09, 04.2012

# 2.7 Multi axis

## 2.7.1 Operating mode Electronic Gear

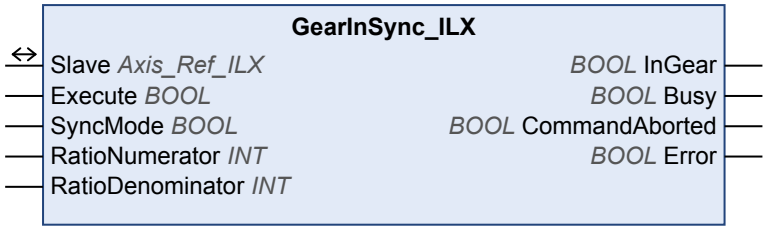
In the operating mode Electronic Gear, movements are carried out according to externally supplied reference value signals. A new position reference value is calculated on the basis of these reference value signals plus an adjustable gear ratio.

### 2.7.1.1 GearInSync_ILX

*Function description*

The function block starts the operating mode Electronic Gear with the method Position Synchronization. In the operating mode Electronic Gear, movements are carried out according to externally supplied reference value signals. A new position value is calculated on the basis of these reference value signals plus an adjustable gear ratio.
In the case of position synchronization without compensation movement, the movement is made synchronously (position synchronicity) with the supplied reference value signals. Reference value signals supplied during an interruption caused by Halt or by an error of error class 1 are not considered.
In the case of position synchronization with compensation movement, the movement is made synchronously (position synchronicity) with the supplied reference value signals. Reference value signals supplied during an interruption caused by Halt or by an error of error class 1 are considered and compensated for. See the product manual for additional information on compensation movements.

*Graphical representation*




*Compatible devices*

ILA

*Inputs/outputs*

The table below shows the inputs/outputs.

| Input/output | Data type | Description |
|---|---|---|
| `Slave` | `Axis_Ref_ILX` | Value range:<br>Initial value:<br>Name of the slave axis |

The table below shows the inputs.

0198441113886, V2.09, 04.2012

| Input | Data type | Description |
|---|---|---|
| SyncMode | BOOL | Value range: FALSE, TRUE<br>Initial value: TRUE<br>FALSE: Position synchronization without compensation movement.<br>TRUE: Position synchronization with compensation movement |
| RatioNumerator | INT | Value range: -32768 ... +32767<br>Initial value: 1<br>Numerator of gear ratio |
| RatioDenominator | INT | Value range: 1 ... 32767<br>Initial value: 1<br>Denominator of gear ratio |

The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| InGear | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>FALSE: Operating mode Electronic Gear not active.<br>TRUE: Operating mode Electronic Gear active. |

"2.5 Basic inputs and outputs"

*Notes*

- This function is not available for ILE, ILS and ILA with multiturn.
- The operating mode must be active for synchronization of reference signals and motor.
- The velocity of the compensation movement (SyncMode = TRUE) is limited by:
  - the maximum current (parameter Settings.I_max).
  - Maximum velocity of the motor.
- Once the operating mode is active, the compensation movement must not exceed the maximum permissible position deviation. If the required compensation movement exceeds the maximum permissible position deviation, a following error is signaled.

*Additional information*

"2.3 PLCopen state diagram"

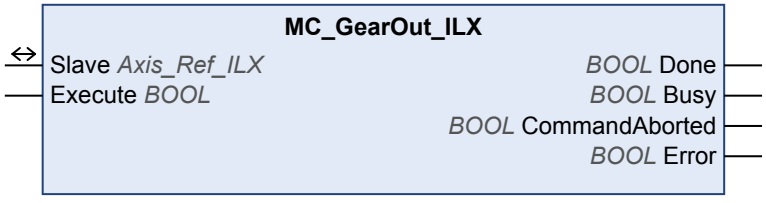
"2.7.1 Operating mode Electronic Gear"

### 2.7.1.2 MC_GearOut_ILX

*Function description*

The function block terminates the operating mode Electronic Gear.

*Graphical representation*




*Compatible devices*

ILA

0198441113886, V2.09, 04.2012

*Inputs/outputs* The table below shows the inputs/outputs.

| Input/output | Data type | Description |
|---|---|---|
| `Slave` | `Axis_Ref_ILX` | Value range:<br>Initial value:<br>Name of the slave axis. |

*Notes*

"2.5 Basic inputs and outputs"

- The function block is only available for ILA.

*Additional information*

"2.3 PLCopen state diagram"

"2.7.1 Operating mode Electronic Gear"

0198441113886, V2.09, 04.2012

# 2.8 Administrative

## 2.8.1 Reading a parameter

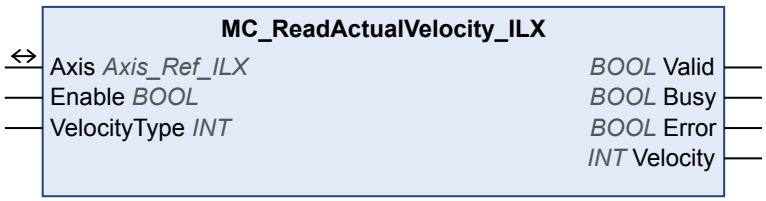
The following functions blocks allow you to read drive parameters such as the actual position or the actual velocity.

An additional function block provides read access to individual parameters of the device. See the product manual for a description of the parameters.

### 2.8.1.1 MC_ReadActualVelocity_ILX

*Function description* The function block is used to read the actual velocity of the motor.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| `VelocityType` | `INT` | Value range: 0 ... 2<br>Initial value: 0<br>Specification of the source of the velocity:<br>• 0: Actual velocity of the motor [$min^{-1}$]<br>• 1: Actual velocity (from profile generator) [$min^{-1}$]<br>• 2: Actual velocity at PTI interface [inc/s]<br>NOTE: Pulse/direction signals or A/B signals are supplied to the drive as reference value signals via the PTI interface (Pulse Train In). |

The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| `Velocity` | `INT` | Value range: 0 ... 65535<br>Initial value: -<br>Unit: [$min^{-1}$]<br>Velocity value of the source selected for the input `VelocityType`. |

"2.5 Basic inputs and outputs"

*Additional information* "2.8.1 Reading a parameter"

### 2.8.1.2 MC_ReadActualPosition_ILX

*Function description* The function block is used to read the actual velocity of the motor.

0198441113886, V2.09, 04.2012

*Graphical representation*


*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| `PositionType` | `INT` | Value range:<br>Initial value: 0<br>Specification of the source of the position:<br>• 0: Actual position of motor [inc]<br>• 1: Reference position (from profile generator) [inc]<br>• 2: Actual position of an external encoder [inc] |

The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| `Position` | `DINT` | Value range: -2147483648 ... 2147483647<br>Initial value: 0<br>Position value in increments of the source selected for the input `PositionType`.<br>ILA: 16384 increments per revolution<br>ILE: 12 increments per revolution<br>ILS: 20000 increments per revolution. |

"2.5 Basic inputs and outputs"

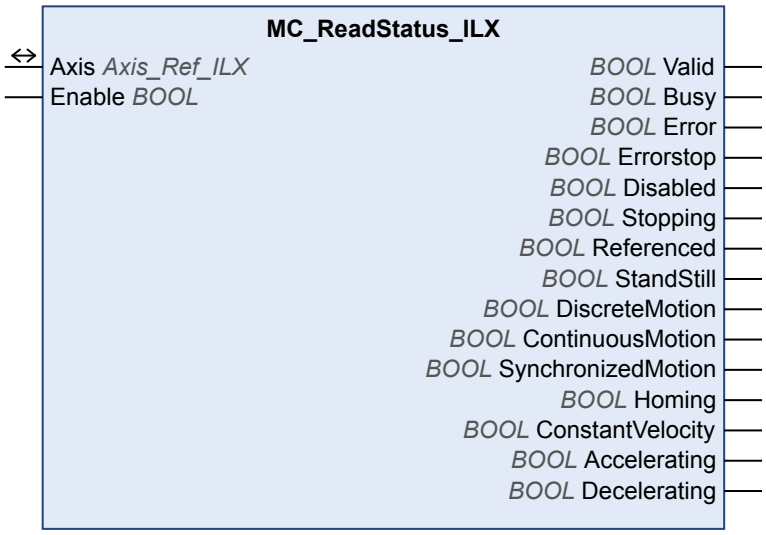
*Additional information* "2.8.1 Reading a parameter"

### 2.8.1.3 MC_ReadStatus_ILX

*Function description* The function block is used to read the current status of the device.

0198441113886, V2.09, 04.2012

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

0198441113886, V2.09, 04.2012

*Inputs/outputs* The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| Errorstop | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>TRUE: The movement has been interrupted by an error. |
| Disabled | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>FALSE: Power stage is enabled.<br>TRUE: Power stage is disabled |
| Stopping | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>TRUE: The function block "2.6.6.1 MC_Stop_ILX" is being executed or the movement is being stopped. |
| Referenced | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>TRUE: The zero point (reference point) is valid. |
| StandStill | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>TRUE: The movement hast been stopped. |
| DiscreteMotion | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>TRUE: The operating mode Profile Position has been started. |
| ContinuousMotion | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>TRUE: The operating mode Profile Velocity has been started. |
| SynchronizedMotion | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>TRUE: A synchronized movement at a constant velocity is performed. (for example, in the operating mode Electronic Gear) |
| Homing | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>TRUE: The operating mode Homing has been started. |
| ConstantVelocity | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>TRUE: A movement at a constant velocity is performed. |
| Accelerating | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>TRUE: The motor accelerates. |
| Decelerating | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>TRUE: The motor decelerates. |

"2.5 Basic inputs and outputs"

*Notes* At any given point in time, the drive is in one of the states: StandStill, Homing, DiscreteMotion, ContinuousMotion, SynchronizedMotion, Stopping, Disabled or ErrorStop. The corresponding output is then TRUE.

*Additional information* "2.3 PLCopen state diagram"

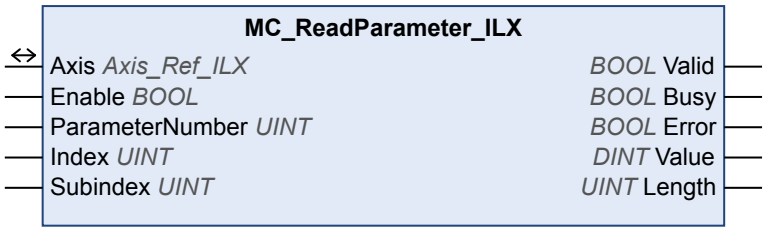
"2.8.1 Reading a parameter"

0198441113886, V2.09, 04.2012

### 2.8.1.4 MC_ReadParameter_ILX

*Function description* The function block reads an object from the device parameter list.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| `ParameterNumber` | `UINT` | Value range: 0 ... 65535<br>Initial value: 1000<br>Number of the parameter:<br>10: Actual velocity.<br>11: Target velocity.<br>1000: Selection via index and subindex. |
| `Index` | `UINT` | Value range: 0 ... 65535<br>Initial value: 0<br>Index of parameter to be read. Only valid if `ParameterNumber` = 1000.<br>See the product manual for an overview of the parameters. |
| `Subindex` | `UINT` | Value range: 0 ... 255<br>Initial value: 0<br>Subindex of parameter to be read. Only valid if `ParameterNumber` = 1000.<br>See the product manual for an overview of the parameters. |

The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| `Value` | `DINT` | Value range: -2147483648 ... 2147483647<br>Initial value: 0<br>Value of the parameter. |
| `Length` | `UINT` | Value range: 0 ... 65535<br>Initial value: 0<br>Length of the parameter in bytes. |

"2.5 Basic inputs and outputs"

*Notes* The function block uses Service Data Objects (SDO) to read the parameter. Therefore, it is strongly recommended not to permanently set the input `Enable` to TRUE. This may cause overload on the field-bus. It is recommended to deactivate the function block when the input `Busy` is set to FALSE.

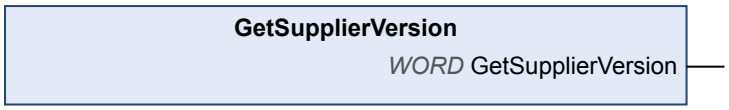
*Additional information* "2.8.1 Reading a parameter"

0198441113886, V2.09, 04.2012

### 2.8.1.5 GetSupplierVersion

*Function description* The function returns the version of the library of the device.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| GetSupplierVersion | WORD | The output provides the version number of the library. Convert the decimal value to hex.<br>Example: GetSupplierVersion = 12368 = $3050_h$ = Version 3.0.5.0 |

*Additional information* "2.8.1 Reading a parameter"

## 2.8.2 Writing a parameter

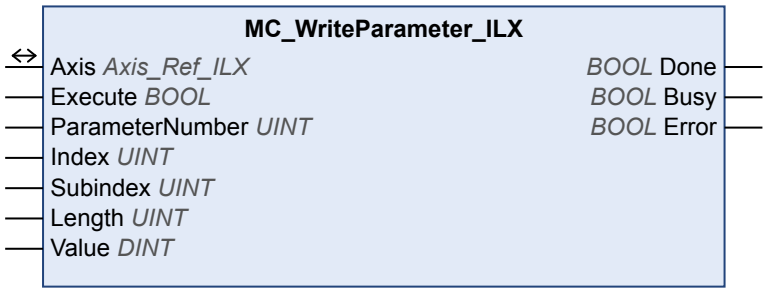
The following function bocks allow you to write drive parameters, for example the values for the acceleration and deceleration ramps.

An additional function block provides write access to individual parameters of the device. See the product manual for a description of the parameters.

### 2.8.2.1 MC_WriteParameter_ILX

*Function description* The function block is used to write a value to a specific parameter.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

0198441113886, V2.09, 04.2012

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| `ParameterNumber` | `UINT` | Value range: 1000<br>Initial value: 1000<br>Reserved. |
| `Index` | `UINT` | Value range: 0 ... 65535<br>Initial value: 0<br>Index of the parameter to be written. See the product manual for a list of the parameters with index and subindex.<br>See the product manual for a list of the parameters with the corresponding CANopen address. |
| `Subindex` | `UINT` | Value range: 0 ... 255<br>Initial value: 0<br>Subindex of the parameter to be written. See the product manual for a list of the parameters with index and subindex.<br>See the product manual for a list of the parameters with the corresponding CANopen address. |
| `Length` | `UINT` | Value range: 0 ... 65535<br>Initial value: 0<br>Length of the parameter to be written in bytes. |
| `Value` | `DINT` | Value range: -2147483648...2147483647<br>Initial value: 0<br>New value to be written to the parameter. |

"2.5 Basic inputs and outputs"

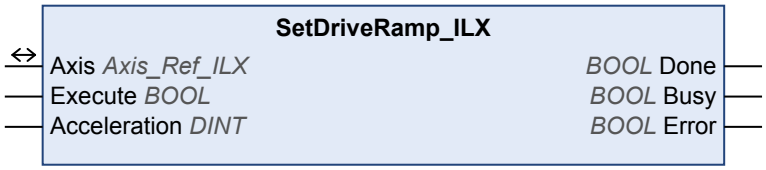
*Notes* If the inputs `ParameterNumber`, `Index` or `Subindex` are changed while `Busy` is TRUE, the function block uses the previous values. The next time the function block is executed, the new values will be used.

*Additional information* "2.8.2 Writing a parameter"

### 2.8.2.2 SetDriveRamp_ILX

*Function description* The function block configures the acceleration ramp and the deceleration ramp of the device.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

0198441113886, V2.09, 04.2012

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| Acceleration | DINT | ILS:<br>Value range: 1 ... 765000<br>Initial value: 600<br>ILE:<br>Value range: 1000 ... 100000<br>Initial value: 2500<br>ILA:<br>Value range: 1 ... 250000<br>Initial value: 600<br>Acceleration ramp and deceleration ramp in revolutions per second [$min^{-1}$/s]. |

"2.5 Basic inputs and outputs"

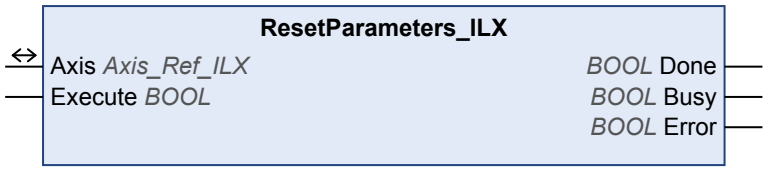
*Notes* Note the following for drives with high external moments of inertia or for highly dynamic applications: The motors regenerate energy during deceleration. The DC bus can absorb a limited amount of energy in the capacitors. Connecting additional capacitors to the DC bus increases the amount of energy that can be absorbed. If the capacity of the capacitors is exceeded, the excess energy must be discharged via internal or external braking resistors. If the energy is not discharged, an overvoltage monitor will shut off the power stage. Overvoltages can be limited by adding a braking resistor with a corresponding braking resistor controller. This converts the regenerated energy to heat energy during deceleration.

*Additional information* "2.8.2 Writing a parameter"

### 2.8.2.3 ResetParameters_ILX

*Function description* This function block restores all parameters to the factory settings.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* "2.5 Basic inputs and outputs"

*Notes* Observe the information provided in chapter .

- The new settings are not saved to the EEPROM. Use "2.8.2.4 StoreParameters_ILX" to save the new settings to the EEPROM.

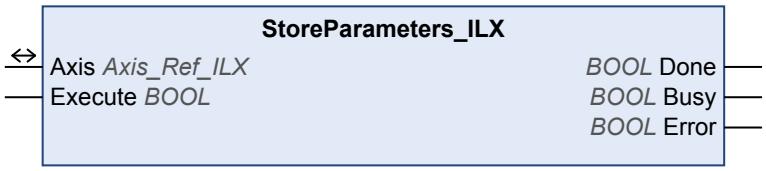
*Additional information* "2.8.2 Writing a parameter"

### 2.8.2.4 StoreParameters_ILX

*Function description* The function block saves the parameter values to the EEPROM.

0198441113886, V2.09, 04.2012

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* "2.5 Basic inputs and outputs"

*Additional information* "2.8.2 Writing a parameter"

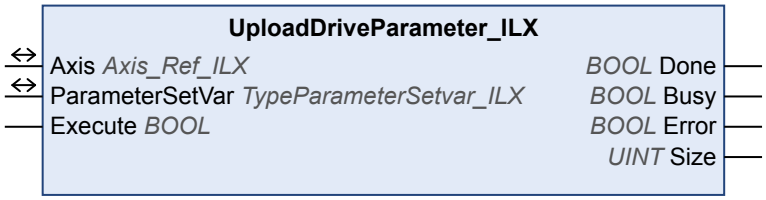
## 2.8.3 Saving and restoring device configuration

Using a function block, you can upload the device configuration from the drive to the controller. A further function block lets you download a device configuration stored on the controller to a drive.

### 2.8.3.1 UploadDriveParameter_ILX

*Function description* The function blocks reads the parameter values that can be modified from the device. See also "2.8.3.2 DownloadDriveParameter_ILX".

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs/outputs.

| Input/output | Data type | Description |
|---|---|---|
| `ParameterSetVar` | `TypeParameterSetvar_ILX` | Value range:<br>Initial value:<br>List of the device parameters. |

The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| `Size` | `UINT` | Value range:<br>Initial value: 0<br>Number of parameters read. In the case of an incorrect upload, the value remains 0. |

"2.5 Basic inputs and outputs"

0198441113886, V2.09, 04.2012

*Notes*

- The function block can only be executed if the drive is in the operating state **3** Switch On Disabled (operating state of drive). To transition to this state, disable the power stage with the function block "2.6.1.1 MC_Power_ILX".
- The two function blocks "2.8.3.2 DownloadDriveParameter_ILX" and "2.8.3.1 UploadDriveParameter_ILX" allow you to save the parameters stored in a device to an identical device without using the commissioning software.

*Additional information* "2.8.3 Saving and restoring device configuration"

### 2.8.3.2 DownloadDriveParameter_ILX

*Function description* The function blocks writes the parameter values that can be modified to the device. Before calling the function block, you must execute "2.8.3.1 UploadDriveParameter_ILX". If not, an error message will be generated.

*Graphical representation*


*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs/outputs.

| Input/output | Data type | Description |
|---|---|---|
| ParameterSetVar | TypeParameterSetvar_ILX | Value range:<br>Initial value:<br>List of device parameters |

The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| Index | UINT | Value range: 0 ... 65535<br>Initial value: 0 ... 255<br>Index of the parameter. See the product manual for an overview of the parameters. |
| Subindex | UINT | Value range:<br>Initial value:<br>Subindex of the parameter. See the product manual for an overview of the parameters. |

"2.5 Basic inputs and outputs"

0198441113886, V2.09, 04.2012

*Notes*

- The function block can only be executed if the drive is in the operating state **3** Switch On Disabled (operating state of drive). To transition to this state, disable the power stage with the function block "2.6.1.1 MC_Power_ILX".
- In order to permanently store the parameters, you must save them to the EEPROM using the function block "2.8.2.4 StoreParameters_ILX".
- The two function blocks "2.8.3.2 DownloadDriveParameter_ILX" and "2.8.3.1 UploadDriveParameter_ILX" allow you to save the parameters stored in a device to an identical device without using the commissioning software.

*Additional information* "2.8.3 Saving and restoring device configuration"

## 2.8.4 Inputs and outputs

The following function blocks allow you to access the digital and analog inputs and outputs of all CAN nodes in the system..

### 2.8.4.1 ConfigureIO_ILX

*Function description* The function block activates and configures the digital inputs and outputs.

*Graphical representation*


*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* "2.5 Basic inputs and outputs"

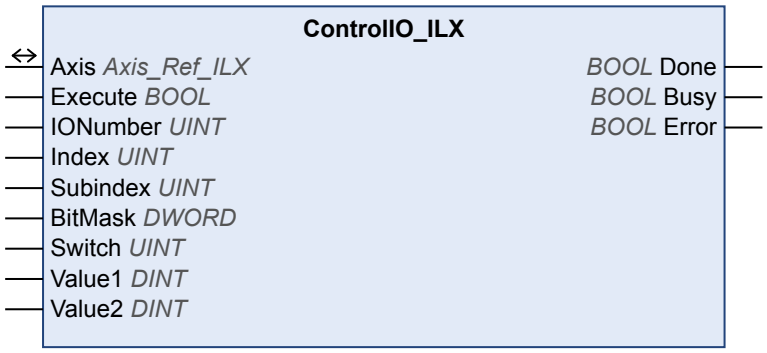
*Additional information* "2.8.4 Inputs and outputs"

### 2.8.4.2 ControlIO_ILX

*Function description* The function block programs the digital inputs and outputs which are configured as programmable in the function block "2.8.4.1 ConfigureIO_ILX".

0198441113886, V2.09, 04.2012

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

0198441113886, V2.09, 04.2012

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| IONumber | UINT | Value range:<br>Initial value:<br>Number of the programmable inputs and outputs:<br>• 0: IO0<br>• 1: IO1<br>• 2: IO2<br>• 3: IO3 |
| Index | UINT | Value range:<br>Initial value:<br>Index of the parameter<br>Programmable input: Index of the parameter whose value is to be written.<br>Programmable output: Index of the parameter whose value is to be read. |
| Subindex | UINT | Value range:<br>Initial value:<br>Subindex of the parameter<br>Programmable input: Subindex of the parameter whose value is to be written.<br>Programmable output: Subindex of the parameter whose value is to be read. |
| BitMask | DWORD | Value range:<br>Initial value:<br>Bit mask.<br>Special case: If the value is "0", the read values remain unchanged. |
| Switch | UINT | Value range:<br>Initial value:<br>Programmable input: Selection of edge<br>Value 0: No change<br>Value 1: Responds to rising edge (value 1)<br>Value 2: Responds to falling edge (value 2)<br>Value 3: Responds to both edges (value 1 and value 2)<br>Programmable output: Comparison criterion<br>Value 0: Read value = comparison value (value 1)<br>Value 1: Read value <> comparison value (value 1)<br>Value 2: Read value < comparison value (value 1)<br>Value 3: Read value > comparison value (value 1) |
| Value1 | DINT | Value range:<br>Initial value:<br>Programmable input: Value is written when edge rises<br>Programmable output: Comparison value for condition |
| Value2 | DINT | Value range:<br>Initial value:<br>Programmable input: Value is written when edge falls<br>Programmable output: Reserved |

"2.5 Basic inputs and outputs"

*Notes* Programmable inputs: The device continuously monitors the status of the inputs for the edge defined via Switch. As soon as the edge is detected, the values of the parameter defined via Index and Subindex are overwritten by the values from Value1 or Value2 under consideration of the bitmask.

0198441113886, V2.09, 04.2012

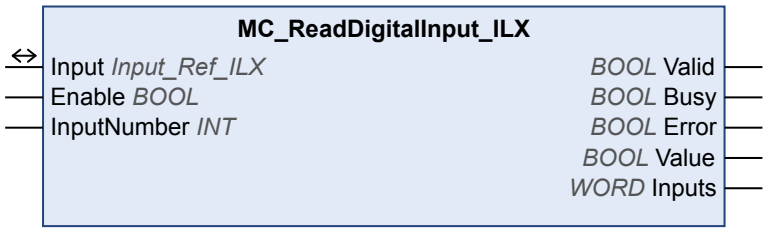
Programmable outputs: The device cyclically reads the value of the parameter defined via `Index` and `Subindex` under consideration of the bitmask. The comparison criterion is set via `Switch`. If the condition is met, the digital output is set.

*Additional information* "2.8.4 Inputs and outputs"

### 2.8.4.3 MC_ReadDigitalInput_ILX

*Function description* Reads the current state of the digital inputs of the drive.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| `InputNumber` | `INT` | Value range: 0 ... 3<br>Initial value: 0<br>Number of the input to be read.<br>Input IO: Bit number<br>IO0: 0<br>IO1: 1<br>IO2: 2<br>IO3: 3 |

The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| `Value` | `BOOL` | Value range: FALSE, TRUE<br>Initial value: FALSE<br>FALSE: Level at selected input is 0 V.<br>TRUE: Level at selected input is 24 V. |
| `Inputs` | `WORD` | Value range: $00_h$ ... $0F_h$<br>Initial value: $00_h$<br>Image of the inputs and outputs as a bit pattern. Bit 0 = first input.<br>Input IO: Bit number<br>IO0: 0<br>IO1: 1<br>IO2: 2<br>IO3: 3 |

"2.5 Basic inputs and outputs"

*Notes* See the product manual for a description of the digital inputs.

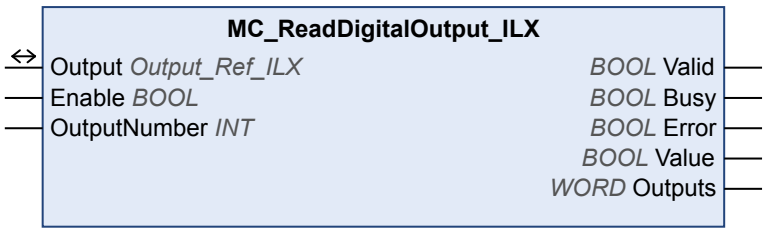
*Additional information* "2.8.4 Inputs and outputs"

0198441113886, V2.09, 04.2012

### 2.8.4.4 MC_ReadDigitalOutput_ILX

*Function description* The function block is used to get the current state of the digital outputs.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| OutputNumber | INT | Value range: 0 ... 3<br>Initial value: 0<br>Number of the output to be read.<br>Output IO: Bit number<br>IO0: 0<br>IO1: 1<br>IO2: 2<br>IO3: 3 |

The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| Value | BOOL | Value range: FALSE, TRUE<br>Initial value: FALSE<br>FALSE: Level at selected output is 0 V.<br>TRUE: Level at selected output is 24 V. |
| Outputs | WORD | Value range: $00_h$ ...$0F_h$<br>Initial value: $00_h$<br>Image of the inputs and outputs as a bit pattern. Bit 0 = first output.<br>Output IO: Bit number<br>IO0: 0<br>IO1: 1<br>IO2: 2<br>IO3: 3 |

"2.5 Basic inputs and outputs"

*Notes* See the product manual for a description of the digital outputs.

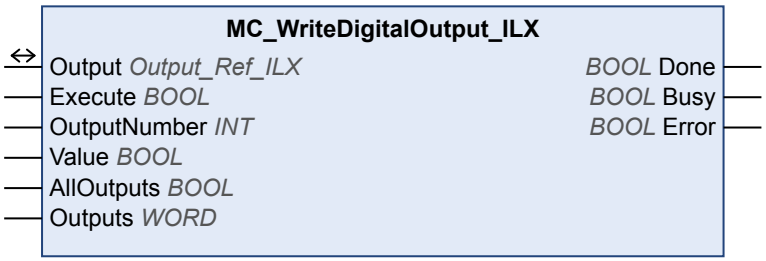
*Additional information* "2.8.4 Inputs and outputs"

### 2.8.4.5 MC_WriteDigitalOutput_ILX

*Function description* The function blocks writes values to the digital outputs.

0198441113886, V2.09, 04.2012

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the inputs.

| Input | Data type | Description |
|---|---|---|
| `OutputNumber` | `INT` | Value range: 0 ... 3<br>Initial value: 0<br>Signal output to which to write.<br>0 = IO0<br>1 = IO1<br>2 = IO2<br>3 = IO3 |
| `Value` | `BOOL` | Value range: FALSE, TRUE<br>Initial value: FALSE<br>FALSE: 0V is written to the selected signal output.<br>TRUE: 24V is written to the selected signal output. |
| `AllOutputs` | `BOOL` | Value range: FALSE, TRUE<br>Initial value:<br>FALSE: The signal output to be written to is set via input `OutputNumber`.<br>TRUE: The signal outputs to be written to are set via input `Output`. |
| `Outputs` | `WORD` | Value range: $0000_h$ ... $000F_h$<br>Initial value: 0<br>0: 0 V is written to the selected signal output.<br>1: 24V is written to the selected signal output.<br>$0000\ 0000\ 0000\ 0001_2$ ($0001_h$) = Signal output 0 (IO0) 24V<br>$0000\ 0000\ 0000\ 0010_2$ ($0002_h$) = Signal output 1 (IO1) 24V<br>$0000\ 0000\ 0000\ 0100_2$ ($0004_h$) = Signal output 2 (IO2) 24V<br>$0000\ 0000\ 0000\ 1000_2$ ($0008_h$) = Signal output 3 (IO3) 24V<br>Example: writing 24 V to all signal outputs:<br>$0000\ 0000\ 0000\ 1111_2$ ($000F_h$) |

"2.5 Basic inputs and outputs"

*Additional information* "2.8.4 Inputs and outputs"

0198441113886, V2.09, 04.2012

## 2.8.5 Error handling

For error handling, each function block has an output `Error` which is set if a synchronous or asynchronous error occurs.

The function block MC_ReadAxisError_xxx is called to analyze the cause of the error. The function block contains the stored error information.

The function block MC_Reset_xxx deletes the error information entered. Future error information can now be stored.

If an additional error occurs, the error information is only stored if no stored error information already exists. If there is still information pertaining to a previous error, the new error message is ignored.

*Table of error numbers*

The table below shows the error numbers of the library. See the product manual for the error numbers of the drive.

| ErrorID hexadecimal | ErrorID decimal | Error class | Description |
|---|---|---|---|
| $0116_h$ | 278 | 1 | Heartbeat or Life Guard error |
| $0134_h$ | 308 | 0 | Parameter out of permissible range |
| $0137_h$ | 311 | 0 | Drive not in operating state**6** Operation Enabled |
| $FF02_h$ | 65282 | - | Server / client command specifier invalid or unknown |
| $FF09_h$ | 65289 | - | No write access, because read object (ro) |
| $FF0A_h$ | 65290 | - | Object does not exist in object dictionary |
| $FF10_h$ | 65296 | - | Data type and parameter length do not match |
| $FF13_h$ | 65299 | - | Subindex not supported |
| $FF14_h$ | 65300 | - | Value range of parameter too large (relevant only for write access) |
| $FF20_h$ | 65312 | 0 | Unknown PLCopen state |
| $FF21_h$ | 65313 | - | Input variable was changed before response was received ("2.8.1.4 MC_ReadParameter_ILX", "2.8.2.1 MC_WriteParameter_ILX") |
| $FF22_h$ | 65314 | 0 | Attempt to interrupt a non-interruptible function block ("2.6.1.1 MC_Power_ILX", "2.6.6.1 MC_Stop_ILX", , "2.6.5.2 MC_SetPosition_ILX") |
| $FF23_h$ | 65315 | 0 | Trigger function is already active |
| $FF24_h$ | 65316 | 0 | PDO timeout |
| $FF25_h$ | 65317 | 0 | Electronic Gear is not active |
| $FF27_h$ | 65319 | 0 | No motor standstill |
| $FF34_h$ | 65332 | 0 | Power stage does not switch to operating state **6** Operation Enabled |
| $FF38_h$ | 65336 | 0 | Parameter list has not yet been read from the device via "2.8.3.1 UploadDriveParameter_ILX". |
| $FF39_h$ | 65337 | 0 | Parameter list and device do not match |
| $FF3B_h$ | 65339 | 0 | Drive is not in operating state**3** Switch On Disabled |
| $FF56_h$ | 65366 | - | Power stage not enabled |

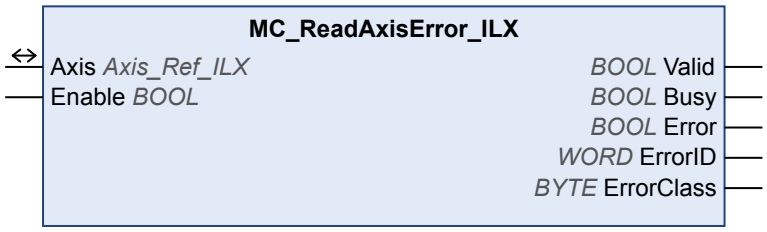
### 2.8.5.1 MC_ReadAxisError_ILX

*Function description*

The function block reads the error information pertaining to the most recent error.

0198441113886, V2.09, 04.2012

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* The table below shows the outputs.

| Output | Data type | Description |
|---|---|---|
| ErrorID | WORD | Value range: $0000_h$ ... $FFFF_h$<br>Initial value: $0000_h$<br>0: No error stored.<br>>0: Stored error number.<br>See the product manual for an overview of the error numbers. |
| ErrorClass | BYTE | Value range: $00_h$ ... $FF_h$<br>Initial value:<br>Error class: The error class determines the response of the device to an error. |

"2.5 Basic inputs and outputs"

*Table of error numbers* The table below shows the error numbers of the library. See the product manual for the error numbers of the drive.

0198441113886, V2.09, 04.2012

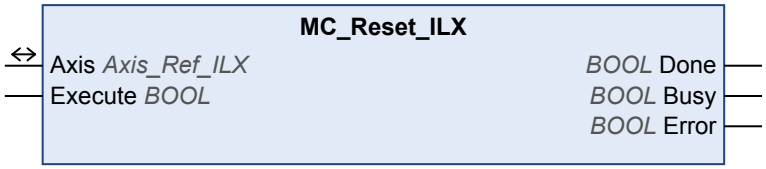
| ErrorID hexadecimal | ErrorID decimal | Error class | Description |
|---|---|---|---|
| $0116_h$ | 278 | 1 | Heartbeat or Life Guard error |
| $0134_h$ | 308 | 0 | Parameter out of permissible range |
| $0137_h$ | 311 | 0 | Drive not in operating state**6** Operation Enabled |
| $FF02_h$ | 65282 | - | Server / client command specifier invalid or unknown |
| $FF09_h$ | 65289 | - | No write access, because read object (ro) |
| $FF0A_h$ | 65290 | - | Object does not exist in object dictionary |
| $FF10_h$ | 65296 | - | Data type and parameter length do not match |
| $FF13_h$ | 65299 | - | Subindex not supported |
| $FF14_h$ | 65300 | - | Value range of parameter too large (relevant only for write access) |
| $FF20_h$ | 65312 | 0 | Unknown PLCopen state |
| $FF21_h$ | 65313 | - | Input variable was changed before response was received ("2.8.1.4 MC_ReadParameter_ILX", "2.8.2.1 MC_WriteParameter_ILX") |
| $FF22_h$ | 65314 | 0 | Attempt to interrupt a non-interruptible function block ("2.6.1.1 MC_Power_ILX", "2.6.6.1 MC_Stop_ILX", , ) |
| $FF23_h$ | 65315 | 0 | Trigger function is already active |
| $FF24_h$ | 65316 | 0 | PDO timeout |
| $FF25_h$ | 65317 | 0 | Electronic Gear is not active |
| $FF27_h$ | 65319 | 0 | No motor standstill |
| $FF34_h$ | 65332 | 0 | Power stage does not switch to operating state **6** Operation Enabled |
| $FF38_h$ | 65336 | 0 | Parameter list has not yet been read from the device via "2.8.3.1 UploadDriveParameter_ILX". |
| $FF39_h$ | 65337 | 0 | Parameter list and device do not match |
| $FF3B_h$ | 65339 | 0 | Drive is not in operating state**3** Switch On Disabled |
| $FF56_h$ | 65366 | - | Power stage not enabled |

*Additional information* "2.8.5 Error handling"

### 2.8.5.2 MC_Reset_ILX

*Function description* The function block is used to acknowledge an error. The error memory is cleared so that it is available for future error messages. If the power stage has been disabled by the automatic error response, it can be enabled again, provided that the cause of the error has been rectified when the error message is acknowledged.

*Graphical representation*




*Compatible devices* ILA, ILE and ILS

*Inputs/outputs* "2.5 Basic inputs and outputs"

0198441113886, V2.09, 04.2012

*Additional information* "2.3 PLCopen state diagram"

"2.8.5 Error handling"

0198441113886, V2.09, 04.2012

## 2.9 Glossary

See chapter "1.5 Standards and terminology" for information on the pertinent standards on which many terms are based. Some terms and abbreviations may have specific meanings with regard to the standards.

*Old Commercial Reference* The Old Commercial Reference is an alphanumerical string that was previously used to identify a product. The Old Commercial Reference can be, for example, a Berger Lahr product identification. Any product that is listed with an Old Commercial Reference also has an up-to-date Commercial Reference which uniquely identifies the product.

*Asynchronous error* Asynchronous errors are signaled without a request. Example of an asynchronous error: Power stage overtemperature.

*Order no.* The Commercial Reference is an alphanumerical string that uniquely identifies a product, for example, in catalogs or on the website. A product can be ordered via its Commercial Reference.

*Limit switch* Switches that signal overtravel of the permissible range of travel.

*Power stage* The power stage controls the motor. The power stage generates current for controlling the motor on the basis of the positioning signals from the controller.

*Fatal error* In the case of fatal error, the product is no longer able to control the motor so that the power stage must be immediately disabled.

*Fault* Fault is a state that can be caused by an error. Further information can be found in the pertinent standards such as IEC 61800-7, ODVA Common Industrial Protocol (CIP).

*Fault reset* A function used to restore the drive to an operational state after a detected error is cleared by removing the cause of the error so that the error is no longer active.

*Error* Discrepancy between a detected (computed, measured or signaled) value or condition and the specified or theoretically correct value or condition.

*Error class* Classification of errors into groups. The different error classes allow for specific responses to errors, for example by severity.

*Device data* The term device data refers to the parameter values of a device. The data is stored in the EEPROM of the device (persistent memory).

*Node guarding* Monitoring of the connection to the slave at an interface for cyclic data traffic.

*Parameter* Device data and values that can be read and set (to a certain extent) by the user.

*Product type* Products are classified by types such as Gearbox, Drive or Motor.

*Product Name* The Product Name consists of the actual name of the product plus some key information on technical details.

*RS485* Fieldbus interface as per EIA-485 which enables serial data transmission with multiple devices.

*Synchronous error* Error signaled by the controller if it is unable to execute a command received from the master.

0198441113886, V2.09, 04.2012

| | |
|---|---|
| *Warning* | If the term is used outside the context of safety instructions, a warning alerts to a potential problem that was detected by a monitoring function. A warning does not cause a transition of the operating state. |
| *Factory setting* | Factory settings when the product is shipped |

0198441113886, V2.09, 04.2012

# 3 Glossary

3

## 3.1 Units and conversion tables

The value in the specified unit (left column) is calculated for the desired unit (top row) with the formula (in the field).

Example: conversion of 5 meters [m] to yards [yd]
5 m / 0.9144 = 5.468 yd

### 3.1.1 Length

| | in | ft | yd | m | cm | mm |
|---|---|---|---|---|---|---|
| **in** | - | / 12 | / 36 | * 0.0254 | * 2.54 | * 25.4 |
| **ft** | * 12 | - | / 3 | * 0.30479 | * 30.479 | * 304.79 |
| **yd** | * 36 | * 3 | - | * 0.9144 | * 91.44 | * 914.4 |
| **m** | / 0.0254 | / 0.30479 | / 0.9144 | - | * 100 | * 1000 |
| **cm** | / 2.54 | / 30.479 | / 91.44 | / 100 | - | * 10 |
| **mm** | / 25.4 | / 304.79 | / 914.4 | / 1000 | / 10 | - |

### 3.1.2 Mass

| | lb | oz | slug | kg | g |
|---|---|---|---|---|---|
| **lb** | - | * 16 | * 0.03108095 | * 0.4535924 | * 453.5924 |
| **oz** | / 16 | - | * $1.942559*10^{-3}$ | * 0.02834952 | * 28.34952 |
| **slug** | / 0.03108095 | / $1.942559*10^{-3}$ | - | * 14.5939 | * 14593.9 |
| **kg** | / 0.45359237 | / 0.02834952 | / 14.5939 | - | * 1000 |
| **g** | / 453.59237 | / 28.34952 | / 14593.9 | / 1000 | - |

### 3.1.3 Force

| | lb | oz | p | N |
|---|---|---|---|---|
| **lb** | - | * 16 | * 453.55358 | * 4.448222 |
| **oz** | / 16 | - | * 28.349524 | * 0.27801 |
| **p** | / 453.55358 | / 28.349524 | - | * $9.807*10^{-3}$ |
| **N** | / 4.448222 | / 0.27801 | / $9.807*10^{-3}$ | - |

### 3.1.4 Power

| | HP | W |
|---|---|---|
| **HP** | - | * 746 |
| **W** | / 746 | - |

0198441113886, V2.09, 04.2012

### 3.1.5 Rotation

| | **$min^{-1}$ (RPM)** | **rad/s** | **deg./s** |
|---|---|---|---|
| **$min^{-1}$ (RPM)** | - | * π / 30 | * 6 |
| **rad/s** | * 30 / π | - | * 57.295 |
| **deg./s** | / 6 | / 57.295 | - |

### 3.1.6 Torque

| | **lb·in** | **lb·ft** | **oz·in** | **Nm** | **kp·m** | **kp·cm** | **dyne·cm** |
|---|---|---|---|---|---|---|---|
| **lb·in** | - | / 12 | * 16 | * 0.112985 | * 0.011521 | * 1.1521 | * $1.129*10^6$ |
| **lb·ft** | * 12 | - | * 192 | * 1.355822 | * 0.138255 | * 13.8255 | * $13.558*10^6$ |
| **oz·in** | / 16 | / 192 | - | * $7.0616*10^{-3}$ | * $720.07*10^{-6}$ | * $72.007*10^{-3}$ | * 70615.5 |
| **Nm** | / 0.112985 | / 1.355822 | / $7.0616*10^{-3}$ | - | * 0.101972 | * 10.1972 | * $10*10^6$ |
| **kp·m** | / 0.011521 | / 0.138255 | / $720.07*10^{-6}$ | / 0.101972 | - | * 100 | * $98.066*10^6$ |
| **kp·cm** | / 1.1521 | / 13.8255 | / $72.007*10^{-3}$ | / 10.1972 | / 100 | - | * $0.9806*10^6$ |
| **dyne·cm** | / $1.129*10^6$ | / $13.558*10^6$ | / 70615.5 | / $10*10^6$ | / $98.066*10^6$ | / $0.9806*10^6$ | - |

### 3.1.7 Moment of inertia

| | **lb·$in^2$** | **lb·$ft^2$** | **kg·$m^2$** | **kg·$cm^2$** | **kp·cm·$s^2$** | **oz·$in^2$** |
|---|---|---|---|---|---|---|
| **lb·$in^2$** | - | / 144 | / 3417.16 | / 0.341716 | / 335.109 | * 16 |
| **lb·$ft^2$** | * 144 | - | * 0.04214 | * 421.4 | * 0.429711 | * 2304 |
| **kg·$m^2$** | * 3417.16 | / 0.04214 | - | * $10*10^3$ | * 10.1972 | * 54674 |
| **kg·$cm^2$** | * 0.341716 | / 421.4 | / $10*10^3$ | - | / 980.665 | * 5.46 |
| **kp·cm·$s^2$** | * 335.109 | / 0.429711 | / 10.1972 | * 980.665 | - | * 5361.74 |
| **oz·$in^2$** | / 16 | / 2304 | / 54674 | / 5.46 | / 5361.74 | - |

### 3.1.8 Temperature

| | **°F** | **°C** | **K** |
|---|---|---|---|
| **°F** | - | (°F - 32) * 5/9 | (°F - 32) * 5/9 + 273.15 |
| **°C** | °C * 9/5 + 32 | - | °C + 273.15 |
| **K** | (K - 273.15) * 9/5 + 32 | K - 273.15 | - |

### 3.1.9 Conductor cross section

| **AWG** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **$mm^2$** | 42.4 | 33.6 | 26.7 | 21.2 | 16.8 | 13.3 | 10.5 | 8.4 | 6.6 | 5.3 | 4.2 | 3.3 | 2.6 |

| **AWG** | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **$mm^2$** | 2.1 | 1.7 | 1.3 | 1.0 | 0.82 | 0.65 | 0.52 | 0.41 | 0.33 | 0.26 | 0.20 | 0.16 | 0.13 |

0198441113886, V2.09, 04.2012

## 3.2 Terms and Abbreviations

See chapter "1.5 Standards and terminology" for information on the pertinent standards on which many terms are based. Some terms and abbreviations may have specific meanings with regard to the standards.

*Asynchronous error* Asynchronous errors are signaled without a request. Example of an asynchronous error: Power stage overtemperature.

*Device data* The term device data refers to the parameter values of a device. The data is stored in the EEPROM of the device (persistent memory).

*Error* Discrepancy between a detected (computed, measured or signaled) value or condition and the specified or theoretically correct value or condition.

*Error class* Classification of errors into groups. The different error classes allow for specific responses to errors, for example by severity.

*Factory setting* Factory settings when the product is shipped

*Fatal error* In the case of fatal error, the product is no longer able to control the motor so that the power stage must be immediately disabled.

*Fault* Fault is a state that can be caused by an error. Further information can be found in the pertinent standards such as IEC 61800-7, ODVA Common Industrial Protocol (CIP).

*Fault reset* A function used to restore the drive to an operational state after a detected error is cleared by removing the cause of the error so that the error is no longer active.

*LED* Light Emitting Diode

*Limit switch* Switches that signal overtravel of the permissible range of travel.

*Node guarding* Monitoring of the connection to the slave at an interface for cyclic data traffic.

*Parameter* Device data and values that can be read and set (to a certain extent) by the user.

*Power stage* The power stage controls the motor. The power stage generates current for controlling the motor on the basis of the positioning signals from the controller.

*RS485* Fieldbus interface as per EIA-485 which enables serial data transmission with multiple devices.

*Synchronous error* Error signaled by the controller if it is unable to execute a command received from the master.

*Warning* If the term is used outside the context of safety instructions, a warning alerts to a potential problem that was detected by a monitoring function. A warning does not cause a transition of the operating state.

0198441113886, V2.09, 04.2012

0198441113886, V2.09, 04.2012

# 4 Index

4



0198441113886, V2.09, 04.2012

0198441113886, V2.09, 04.2012