acontis technologies GmbH

SOFTWARE

EC-Motion-Advanced

User Manual

Version 3.2

Edition: September 24, 2025

Contents

1 Introduction

1.1 General

The goal of a motion control library is the hardware independent implementation of movement commands. It should be easy to use in terms of installation, maintenance and application programming. Furthermore an efficient design increases the understanding and consistency of the code. Future extensions are possible without any problems and the scope of the library is not mandatory but sufficiently complete.

EC-Motion-Advanced is a motion control solution supporting EtherCAT. The main component is an abstractation of an axis object which could be a drive controlled over EtherCAT or a virtual drive. With the help of motion control function blocks designed as C++ classes orders are commanded to an axis. One distinguishes between motion and administrative function blocks. All internal tasks such as trajectory generation, interpolation and state machines are hidden from the user in this way.

EC-Motion-Advanced is targeted to work in conjunction with the acontis EC-Master (EtherCAT Master library) but the library is not mandatory.

For optimal use of EC-Motion-Advanced, it is highly recommended to familiarize yourself with the EC-Master user manual and the User Manual of your drive.

1.1.1 The EC-Motion-Advanced - Features

EC-Motion-Advanced supports the CANopen device profile CiA® 402 for drives and motion control. The physical communication channel is EtherCAT and the CiA 402 protocol is mapped to EtherCAT according to IEC 61800-7-300 as “drive profiles”.

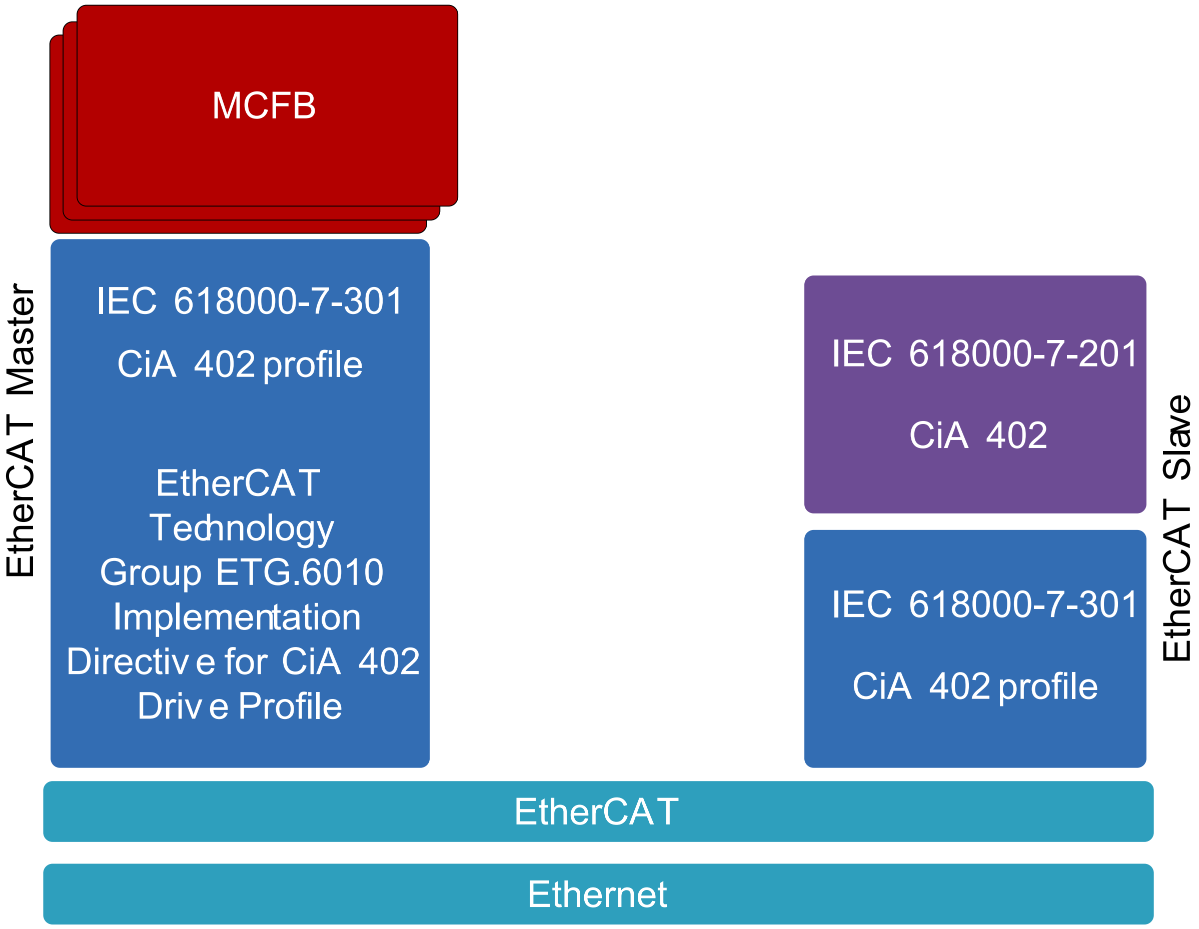


Fig. 1.1: EtherCAT servo drive profiles

In addition EC-Motion-Advanced provides virtual axes with no hardware or protocol in the background.

1.2 Open Source Software

EC-Motion-Advanced contains no free open source software.

1.3 Versioning

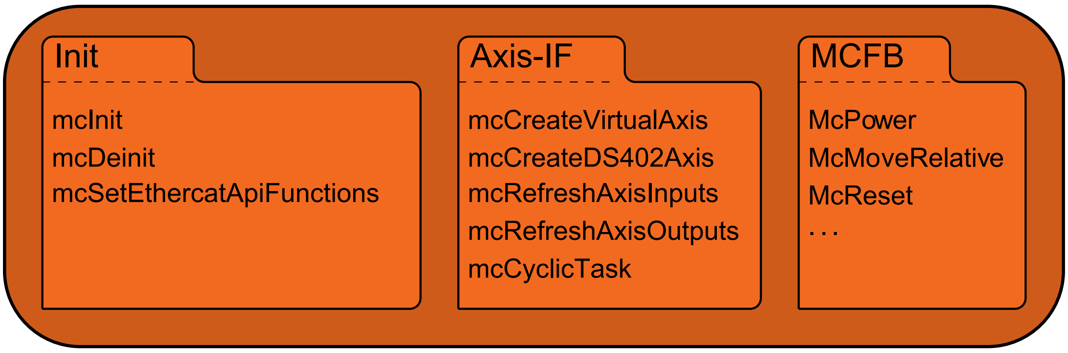
EC-Motion-Advanced follows the same versioning scheme as the EC-Master: **V***MAJOR***.** *MINOR***.** *SERVICEPACK***.** *BUILD* (e.g. V3.2.1.04).

The libraries are binary compatible by unchanged *MAJOR* and *MINOR* digits. If *SERVICEPACK* increments, *BUILD* restarts with 01. *BUILD* 99 is reserved for test builds that have not been officially released for productive usage.

2 Getting Started

2.1 EC-Motion-Advanced Architecture

The EC-Motion-Advanced library is implemented in C++ and offers a C++ API. The library exports an opaque class *AxisRef* which serves as an abstraction level to the hardware. The motion control function blocks are implemented as callable classes with writable input and readable output parameters. Most function blocks have an *AxisRef* as input parameter. Furthermore functions for initialization, deinitialization, axis creation and cyclic tasks are exported.



2.2 Running EcMasterDemoMotionAdvanced

The demo application *EcMasterDemoMotionAdvanced* is part of the delivery as source code and “out of the box” application. It serves as a reference implementation of a motion control application. The example located in the folder *Example* can be extended or used as a starting point for an own motion control application.

The demo depends on the EC-Master product (which is not included in the delivery). In order to control the drives externally the EC-Engineer is also needed. Figure Fig. 2.1 shows the whole architecture used to control three servo drives that are connected to the EtherCAT fieldbus.

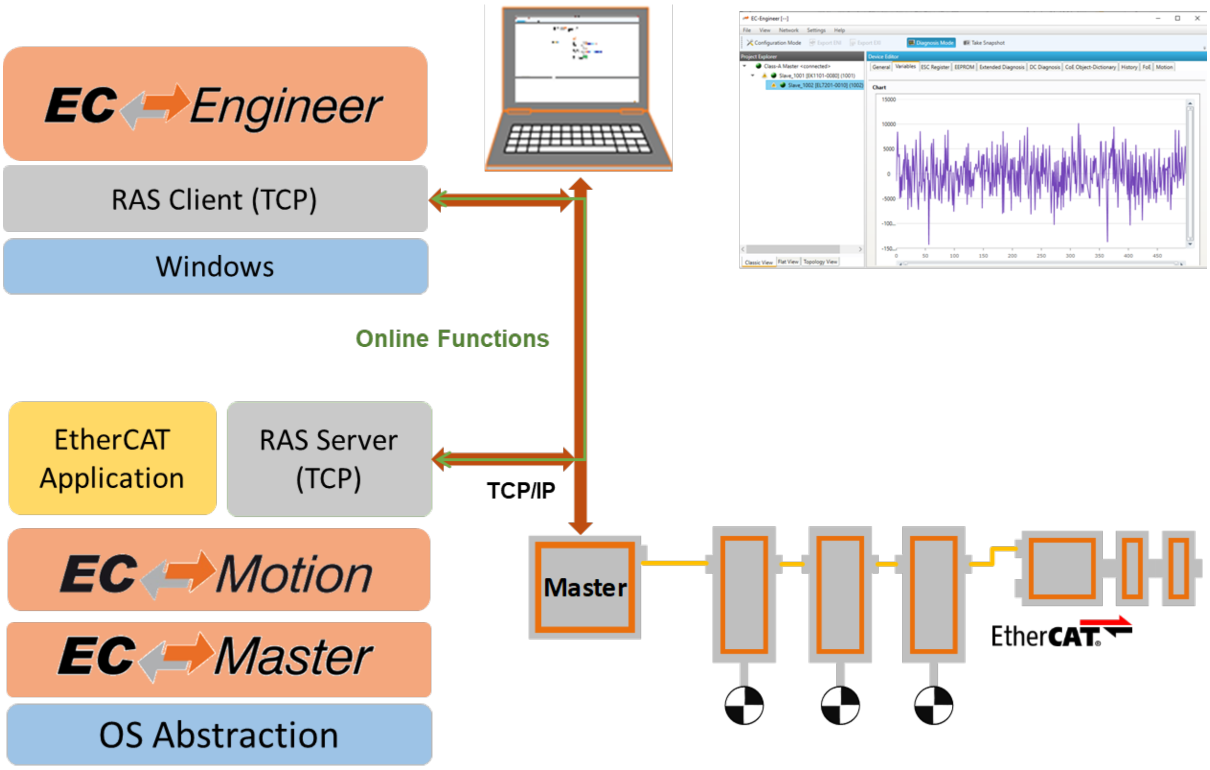


Fig. 2.1: Building blocks of EC-Motion-Advanced control system

Because motion control applications run with small cycle times a real-time operating system is highly recommended to use.

Start the EcMasterDemoMotionAdvanced from the command line to put the EtherCAT network into operation and to initiate the Remote Access Server (RAS server). Furthermore start the EC-Engineer or EC-STA under Windows to send motion commands via TCP/IP sockets to the demo application.

```
> EcMasterDemoMotionAdvanced DemoConfig.xml
```

2.2.1 Command line parameters

```
EcMasterDemoMotionAdvanced <LinkLayer> [-cfg DemoConfigFileName] [-f ENI-
FileName] [-t time] [-b cycle time] [-a affinity] [-v level] [-
perf [level]] [-log prefix [msg cnt]] [-lic key] [-oem key] [-
maxbusslaves cnt] [-flash address] [-printvars] [-sp [port]] [-ctloff] [-
rec [prefix [frame cnt]]] [-junctionred] [-
dcmmode <mode> [<synctocyclestart>]] [-dcmlog]
```

```
EcMasterDemoMotionAdvanced <DemoConfigFileName>
```

The parameters are as follows:

-f `<ENI-FileName>`
Path to ENI file

-cfg `<DemoConfigFileName>`
Path to demo configuration file.

-t `<time>`
Running duration in msec. When the time expires the demo application exits completely.

<time>
Time in msec, 0 = forever (default = 120000)

-b `<cycle time>`

Specifies the bus cycle time. Defaults to 1000 µs (1 ms).

<cycle time>
Bus cycle time in µsec

-a `<affinity>`

The CPU affinity specifies which CPU the demo application ought to use.

<affinity>
0 = first CPU, 1 = second, …

-v `<level>`

The verbosity level specifies how much console output messages will be generated by the demo application. A high verbosity level leads to more messages.

<level>
Verbosity level: 0=off (default), 1..n=more messages

-perf `[<level>]`

Enable max. and average time measurement in µs for all EtherCAT jobs (e.g. ProcessAllRxFrames).

<level>
Depending on level the performance histogram can be activated as well.

-log `<prefix> [<msg cnt>]`

Use given file name prefix for log files.

<prefix>

<msg cnt>
Messages count for log buffer allocation

-lic `<key>`

Set License key.

<key>
License key string

-oem `<key>`

Use OEM key

<key>
64 bit OEM key.

-flash `<address>`

Flash outputs

<address>
0=all, >0 = slave station address

-sp `[<port>]`

If platform has support for IP Sockets, this command-line option enables the Remote API Server to be started. The Remote API Server is going to listen on TCP Port 6000 (or port parameter if given) and is available for connecting Remote API Clients.

<port>
RAS server port

-rec `[<prefix> [<frame cnt>]]`

Packet capture file recording

<prefix>
File name prefix

<frame cnt>
Frame count for log buffer allocation

-dcmmode `<mode> [<synctocyclestart>]`
Set DCM mode

<mode>
off | busshift | mastershift | masterrefclock | linklayerrefclock

<synctocyclestart>
Sync to cycle start: 0 = disabled (default), 1 = enabled

-dcmlog
Enable DCM logging (default: disabled)

-ctloff
Disable DCM control loop for diagnosis

2.2.2 Configuration

The EC-Engineer tool can create the configuration file. In oder to activate the motion tabs inside EC-Engineer the *Motion Mode* view has to be selected (see also Figure Fig. 2.2).

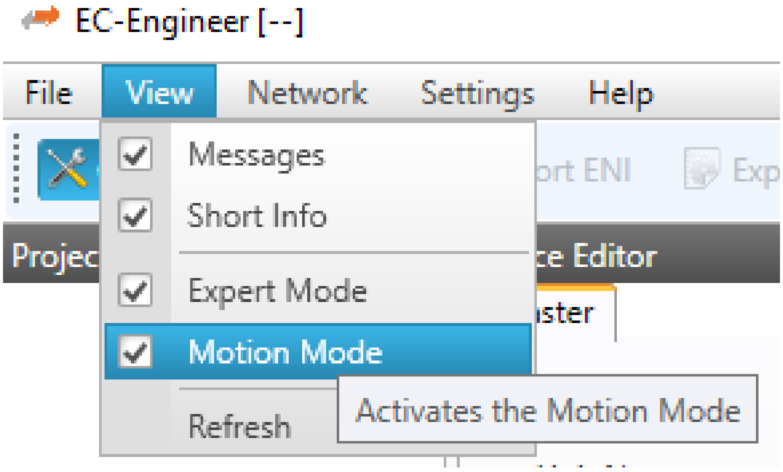
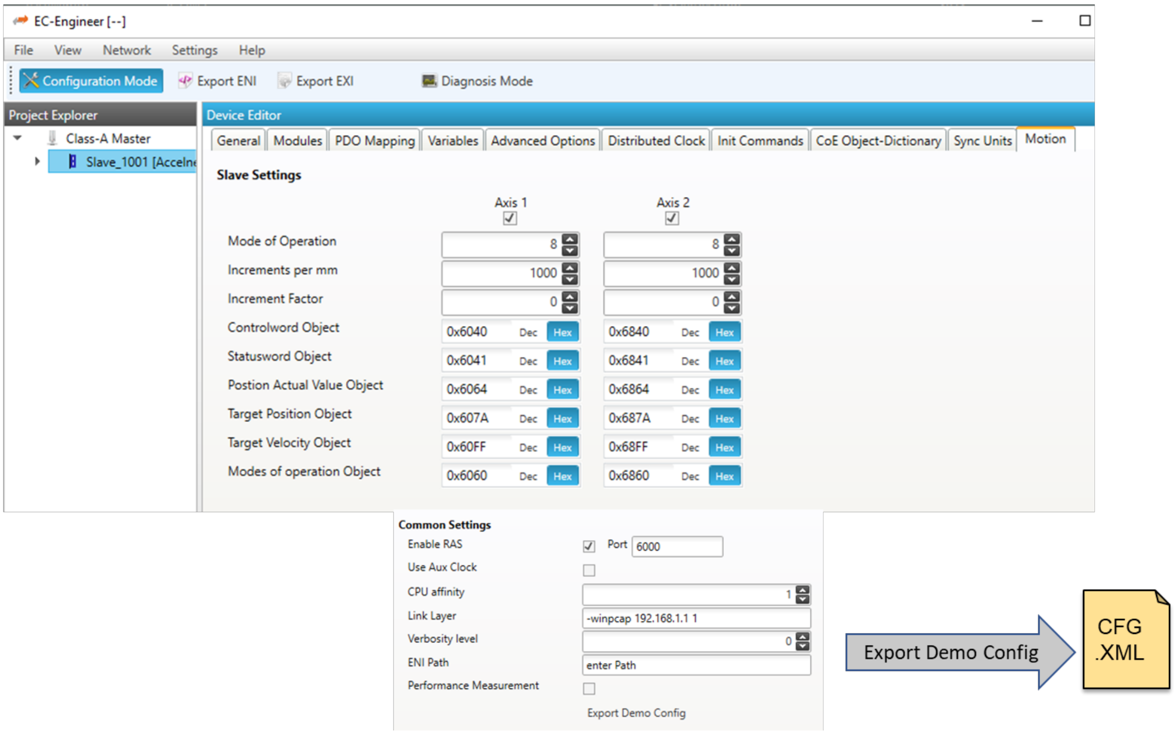


Fig. 2.2: Activate Motion Mode

After the *Motion Mode* was activated there is a motion tab inside the Class-A or Class-B master and each EtherCAT slave. Here the different parameters for the drives and the EtherCAT network can be set and finally exported. Figure Fig. 2.3 shows an example.



Slave Settings

	Axis 1 ☑	Axis 2 ☑
Mode of Operation	8	8
Increments per mm	1000	1000
Increment Factor	0	0
Controlword Object	0x6040 Dec Hex	0x6840 Dec Hex
Statusword Object	0x6041 Dec Hex	0x6841 Dec Hex
Postion Actual Value Object	0x6064 Dec Hex	0x6864 Dec Hex
Target Position Object	0x607A Dec Hex	0x687A Dec Hex
Target Velocity Object	0x60FF Dec Hex	0x68FF Dec Hex
Modes of operation Object	0x6060 Dec Hex	0x6860 Dec Hex

Common Settings

Enable RAS	☑ Port 6000
Use Aux Clock	□
CPU affinity	1
Link Layer	-winpcap 192.168.1.1 1
Verbosity level	0
ENI Path	enter Path
Performance Measurement	□
	Export Demo Config

Fig. 2.3: Export configuration parameter

Afterwards the parameters can be adjusted directly within the xml file when the configuration changes or to customize the drives. The entries of the configuration file are as follows:

`Config/Common/RASEnabled`
Start the RAS server or not when the application starts. If the RAS server is not enabled a remote control is not possible.

`Config/Common/RASPort`
The TCP/IP socket port number for the RAS server. Default is 6000.

`Config/Common/BusCycleTime`
Cycle period of the cyclic task in microseconds. Default is 1000.

`Config/Common/AuxClk`
Auxiliary clock period in microseconds. Default is 1000. Set to 0 if the auxillary clock (Hardware interrupt timer) is not supported for this particular platform.

`Config/Common/CpuAffinity`
Index of the CPU on which the various threads are running. Default is 0 (first CPU). 1 is CPU2, 2 is CPU3, …

`Config/Common/LinkLayer`
Initialization string for the LinkLayer driver. Please see the EC-Master manual for details.

`Config/Common/ENIFileName`
Path to the EtherCAT Network Information (ENI) file.

`Config/Common/VerbosityLevel`
Verbosity level for log messages. Default is 2.

`Config/Common/DemoDuration`
How long in seconds the program should run.

`Config/Common/PerfMeasurement`
Enable tracing of performance related data. Default is 1.

`Config/MotionDemo/NoDCMBusShift`
Disable Distributed Clocks Master (DCM) bus shift controller. Default is 0 for enable DCM.

`Config/MotionDemo/DCMCtlSetVal`
DCM controller set value in nanoseconds.

`Config/MotionDemo/CmdMode`
If the value is zero, the demo application runs standalone and turns the configured axes forward and backward using MCFB McMoveRelative.

If the value is one, the motion is remotely commanded with either EC-Engineer or EC-STA.

`Config/MotionDemo/Drive[N]/Address`
EtherCAT station address of drive.

`Config/MotionDemo/Drive[N]/OperationMode`
Set the mode of operation for axis. Please read the manual of your servo drive controller for detailed information about the different drive operating modes.

`Config/MotionDemo/Drive[N]/IncPerMM`
Increments per physical unit [u].

`Config/MotionDemo/Drive[N]/IncFactor`
Internal resolution factor of the motion kernel. When new positions are computed using fixpoint mathematics, the internally used positions are scaled by two to the power of the given value. So if you see poor resolution (e.g. drive will not move at very low velocities), increment this value one by one.

`Config/MotionDemo/Drive[N]/`
`Idx[Status|Control|Mode|ModeDisplay|Posact|Targetpos|Targetvel]`
PDO index in the format “0xXXXX:0xYY” of the corresponding variable in the PDO mapping. If an entry is not present, the default value according to DS402 is used. The relevant PDO’s and PDO-variables should be included inside the EtherCAT configuration tool and exported to the ENI file.

`Config/MotionDemo/Drive[N]/DriveProfile`
Define the drive profile. If set to “DS402”, the drive acts like a DS402 drive. If set to “VIRTUAL”, no drive profile is selected and a virtual axis is created with no hardware connection.

`Config/MotionDemo/Drive[N]/CoeIdxOpMode`
If the PDO variable ‘Mode of Operation’ is not part of the EtherCAT network configuration, the index in the CoE object dictionary could be set here to transfer the given operation mode via SDO request.

`Config/MotionDemo/Drive[N]/[Vel|Acc|Dec|Jerk|Distance]`
Maximal values for velocity, acceleration, decceleration, jerk and distance used during the standalone application run (see also ‘CmdMode’).

2.3 Compiling the EcMasterDemoMotionAdvanced

The application EcMasterDemoMotionAdvanced is delivered as binary and also as source code. So the demo can be adjusted to fit the given hardware configuration. E.g. by default four axes are supported by EcMasterDemoMotionAdvanced. This number can be adjusted by changing the define DEMO_MAX_NUM_OF_AXIS within the source code.

The following main rules can be used to generate the example applications for all operating systems.

- `<OS>` is a placeholder for the operating system used.
- `<ARCH>` for the architecture. If different architectures are supported.

2.3.1 EC-Motion-Advanced Software Development Kit (SDK)

The EC-Motion-Advanced development kit is needed to write applications based on the EC-Motion-Advanced core. The EC-Motion-Advanced core is shipped as a library which is linked together with the application.

The following components are supplied together with an SDK:

```
<InstallPath>/Bin
<InstallPath>/Doc
<InstallPath>/SDK
<InstallPath>/Examples
<InstallPath>/SDK/INC
<InstallPath>/SDK/LIB
<InstallPath>/SDK/FILES
<InstallPath>/Sources/Common
```

/Bin
Executables containing the EC-Motion-Advanced core.

/Doc
Documentation

/Examples
Example applications as source code.

/SDK
EtherCAT Software Development Kit containing libraries and header files to build C/C++-applications.

/SDK/INC:
Header files to be included with the application

/SDK/LIB:
Libraries to be linked with the application

/SDK/FILES:
Additional files for platform integration (e.g. Windows CE registry files)

/Sources/Common:
Shared .cpp-files

2.3.2 Include search path

The header files are located in the following directories:

```
<InstallPath>/SDK/INC
<InstallPath>/SDK/INC/<OS>/<ARCH>
<InstallPath>/Sources/Common
```

2.3.3 Library

The library is located in the following directory:

```
<InstallPath>/SDK/LIB/<OS>/<ARCH>
```

3 Application programming interface, reference

The main header file to include is `EcMotionAdvanced.h`. There the API function prototypes, error codes and motion control function blocks can be found. Structure definitions are specified in `EcMotionDef.h` which is included directly by `EcMotionAdvanced.h` and therefor must not be included by the application.

3.1 Functions for initialization and deinitialization

3.1.1 mcInit

EC_T_DWORD **`mcInit`** (EC_T_DWORD dwInstanceId, const *MC_T_INIT_PARMS* &rParms)

Initialize motion control library.

This function has to be called prior to any other function of EC-Motion-Advanced.

Parameters

- **`dwInstanceId`** – Identification of instance.
- **`rParms`** – Parameter for initialization.

Returns

If initialization was successful, EC_E_NOERROR is returned. Otherwise a corresponding error code is given.

struct **`MC_T_INIT_PARMS`**

Public Members

EC_T_DWORD **`dwSignature`**

[in] Set to MC_SIGNATURE

EC_T_DWORD **`dwSize`**

[in] Set to sizeof(MC_T_INIT_PARMS)

EC_T_LOG_PARMS **`LogParms`**

[in] Parameters for logging

EC_T_DWORD **`dwMaxMotionAxes`**

[in] Maximal number of motion axes

EC_T_DWORD **`dwMaxCamTables`**

[in] Maximal number of CAM tables

struct **`EC_T_LOG_PARMS`**

Public Members

EC_T_DWORD **dwLogLevel**
[in] Log level. See EC_LOG_LEVEL_…

EC_PF_LOGMSGHK **pfLogMsg**
[in] Log callback function called on every message

struct _EC_T_LOG_CONTEXT ***pLogContext**
[in] Log context to be passed to log callback

For further information about the treatment of log messages see the user manual of EC-Master.

Example

```
MC_T_INIT_PARMS oMcInitParms;
EC_T_DWORD dwRetVal;
OsMemset(&oMcInitParms, 0, sizeof(MC_T_INIT_PARMS));
oMcInitParms.dwSignature = MC_SIGNATURE;
oMcInitParms.dwSize = sizeof(MC_T_INIT_PARMS);
oMcInitParms.dwMaxMotionAxes = 4;
dwRetVal = mcInit(INSTANCE_DEFAULT, oMcInitParms);
```

3.1.2 mcSetEthercatApiFunctions

EC_T_DWORD **mcSetEthercatApiFunctions** (const *MC_T_INIT_ECAT_PARMS* &rEcatParms)
Publish API functions of EtherCAT master to motion control library in order to communicate with master and network.

Parameters
rEcatParms – API function pointers.

Returns
EC_E_NOERROR is returned.

struct **MC_T_INIT_ECAT_PARMS**

Public Members

MC_PF_GET_SLAVE_STATE **pfnGetSlaveState**
[in] Function to get operation state of slave.

MC_PF_COE_SDO_DOWNLOAD_REQ **pfnCoeSdoDownload**
[in] Function to download value to slave via SDO.

typedef EC_T_DWORD (***MC_PF_GET_SLAVE_STATE**)(EC_T_DWORD dwInstanceId, EC_T_DWORD dwSlaveId, EC_T_WORD *pwCurrDevState, EC_T_WORD *pwReqDevState)
Determine the current operation state of desired slave device.

Parameters

- **dwInstanceId** – [in] Identification of instance.
- **dwSlaveId** – [in] Identification of slave.

- **`pwCurrDevState`** – [out] Current slave state.
- **`pwReqDevState`** – [out] Requested slave state

Returns

EC_E_NOERROR if function call was successful. Otherwise an error code is returned.

typedef EC_T_VOID (***`MC_PF_COE_SDO_DOWNLOAD_REQ`**)(EC_T_DWORD dwInstanceId, EC_T_DWORD dwSlaveId, EC_T_WORD wObjIdx, EC_T_BYTE byObjSubidx, EC_T_BYTE *pbyData, EC_T_DWORD dwDataLen, EC_T_BOOL *pbFinished, EC_T_DWORD *pdwResult)

Request a download to CoE dictionary of given slave.

Parameters

- **`dwInstanceId`** – [in] Identification of instance.
- **`dwSlaveId`** – [in] Identification of slave.
- **`wObjIdx`** – [in] Index of object dictionary.
- **`byObjSubidx`** – [in] Subindex of object dictionary.
- **`pbyData`** – [in] Pointer to data for download.
- **`dwDataLen`** – [in] Length of data for download.
- **`pbFinished`** – [out] Indicates when the download has finished with or without any error.
- **`pdwResult`** – [out] Error code when download was erroneous.

Example

```
MC_T_INIT_ECAT_PARMS oMcInitEcatParms;
EC_T_DWORD dwRetVal;
OsMemset(&oMcInitEcatParms, 0, sizeof(MC_T_INIT_ECAT_PARMS));
oMcInitEcatParms.pfnGetSlaveState = emGetSlaveState;
dwRetVal = mcSetEthercatApiFunctions(oMcInitEcatParms);
```

An example for the definition of a SDO download function can be found in the demo application.

3.1.3 mcDeinit

EC_T_DWORD **`mcDeinit`**(EC_T_DWORD dwInstanceId)

Deinitialize motion control library. During the function also all created axes belonging to the given instance are destroyed. So they cannot be used any longer in any motion function block for example.

Parameters

`dwInstanceId` – Identification of instance.

Returns

If function call was successful, EC_E_NOERROR is returned. Otherwise a corresponding error code is given.

Example

```
EC_T_DWORD dwRetVal;
dwRetVal = mcDeinit(INSTANCE_DEFAULT);
```

3.2 Create axis

Almost all motion control function blocks have an `AxisRef` as input. It is an abstraction of the hardware and its drive.

class **`AxisRef`**

Subclassed by AxisRefBase

An object of this class must be created by one of the next functions.

3.2.1 mcCreateVirtualAxis

EC_T_DWORD **`mcCreateVirtualAxis`** (
const *MC_T_AXIS_PARMS* &rAxisParms,
AxisRef *&pAxisRef
)

Create a virtual axis.

Parameters

- **`rAxisParms`** – [in] Parameter set for axis.
- **`pAxisRef`** – [out] Storage to save pointer to new created axis reference.

Returns

When creation of axis reference was successful, EC_E_NOERROR or EC_E_AXIS_NOT_OPERATIONAL is returned. The latter means that an axis object has been created but it is not operational and cannot be used, e.g. to move a drive. In case the creation was not successful a corresponding error identification is given.

struct **`MC_T_AXIS_PARMS`**

Public Members

EC_T_DWORD **`dwInstanceId`**

[in] Identification of instance

EC_T_DWORD **`dwTaskId`**

[in] Identification of task

EC_T_DWORD **`dwCycleTime`**

[in] [us] Cycle time axis is served cyclically

EC_T_DWORD **`dwIncPerMM`**

[in] Increments per mm

EC_T_DWORD **`dwIncFactor`**

[in] Internal position values are 2^x times bigger

EC_T_LREAL **lfMaxVelocity**
: [in] Maximal velocity of axis (infinite when set to zero)

EC_T_LREAL **lfMaxAcceleration**
: [in] Maximal acceleration/deceleration of axis (infinite when set to zero)

EC_T_LREAL **lfMaxJerk**
: [in] Maximal jerk of axis (infinite when set to zero)

EC_T_DWORD **dwVelocityGain**
: [in] CSV-Mode: Velocity Gain, CSP-Mode: Velocity Gain for Feed Forward Object 0x60B1

EC_T_DWORD **dwTorqueGain**
: [in] CSP and CSV-Mode: Torque Gain for Feed Forward Object 0x60B2

EC_T_DWORD **dwMaxMoveCmds**
: [in] Maximal number of move commands that can be queued

Example

```
AxisRef* S_pAxis = EC_NULL;
EC_T_DWORD dwRetVal = EC_E_ERROR;
MC_T_AXIS_PARMS oAxisParms;
OsMemset(&oAxisParms, 0, sizeof(MC_T_AXIS_PARMS));
oAxisParms.dwCycleTime = 1000;
oAxisParms.dwIncPerMM = 10000;
oAxisParms.dwMaxMoveCmds = 8;
dwRetVal = mcCreateVirtualAxis(oAxisParms, S_pAxis);
```

3.2.2 mcCreateDS402Axis

EC_T_DWORD **mcCreateDS402Axis** (
const *MC_T_AXIS_PARMS* &rAxisParms,
const *MC_T_AXIS_IO_ENDPOINT* &rAxisIo,
const *MC_T_AXIS_ECAT_PARMS* &rAxisEcatParms,
AxisRef *&pAxisRef
)

Create a DS402 axis.

Parameters

- **rAxisParms** – [in] Parameter set for axis.
- **rAxisIo** – [in] EtherCAT IO endpoints of axis.
- **rAxisEcatParms** – [in] EtherCAT specific parameters.
- **pAxisRef** – [out] Storage to save pointer to new created axis reference.

Returns

When creation of axis reference was successful, EC_E_NOERROR or EC_E_AXIS_NOT_OPERATIONAL is returned. The latter means that an axis object has been created but it is not operational and cannot be used, e.g. to move a drive. In case the creation was not successful a corresponding error identification is given.

See also:

MC_T_AXIS_PARMS

struct **MC_T_AXIS_IO_ENDPOINT**

Public Members

EC_T_DWORD **dwBitOffsControlWord**
[in] Bit offset within output process data image for control word of axis (type EC_T_WORD)

EC_T_DWORD **dwBitSizeControlWord**
[in] Bit size of control word in process data image

EC_T_DWORD **dwBitOffsTargetPosition**
[in] Bit offset within output process data image for target position of axis (type EC_T_SDWORD); in Cyclic Synchronous Position mode the value is always interpreted as an absolute value

EC_T_DWORD **dwBitSizeTargetPosition**
[in] Bit size of target position in process data image

EC_T_DWORD **dwBitOffsTargetVelocity**
[in] Bit offset within output process data image for target velocity of axis (type EC_T_SDWORD)

EC_T_DWORD **dwBitSizeTargetVelocity**
[in] Bit size of target velocity in process data image

EC_T_DWORD **dwBitOffsVelocityOffset**
[in] Bit offset within output process data image for velocity offset of axis (type EC_T_SDWORD)

EC_T_DWORD **dwBitSizeVelocityOffset**
[in] Bit size of velocity offset in process data image

EC_T_DWORD **dwBitOffsTargetTorque**
[in] Bit offset within output process data image for target torque of axis (type EC_T_SDWORD)

EC_T_DWORD **dwBitSizeTargetTorque**
[in] Bit size of target torque in process data image

EC_T_DWORD **dwBitOffsTorqueOffset**
[in] Bit offset within output process data image for torque offset of axis (type EC_T_SDOWRD)

EC_T_DWORD **dwBitSizeTorqueOffset**
[in] Bit size of torque offset in process data image

EC_T_DWORD **dwBitOffsModeOfOperation**
[in] Bit offset within output process data image for mode of operation of axis (type EC_T_BYTE)

EC_T_DWORD **dwBitSizeModeOfOperation**
[in] Bit size of mode of operation in process data image

EC_T_DWORD **dwBitOffsDigitalOutputs**
[in] Bit offset within output process data image for digital outputs of axis (type EC_T_DWORD)

EC_T_DWORD **dwBitSizeDigitalOutputs**
[in] Bit size of digital outputs in process data image

EC_T_DWORD **dwBitOffsStatusWord**
[in] Bit offset within input process data image for status word of axis (type EC_T_WORD)

EC_T_DWORD **dwBitSizeStatusWord**
[in] Bit size of status word in process data image

EC_T_DWORD **dwBitOffsActualPosition**
[in] Bit offset within input process data image for actual position of axis (type EC_T_SDWORD)

EC_T_DWORD **dwBitSizeActualPosition**
[in] Bit size of actual position in process data image

EC_T_DWORD **dwBitOffsActualTorque**
[in] Bit offset within input process data image for actual torque of axis (type EC_T_SWORD)

EC_T_DWORD **dwBitSizeActualTorque**
[in] Bit size of actual torque in process data image

EC_T_DWORD **dwBitOffsModeOfOperationDisplay**
[in] Bit offset within input process data image for mode of operation display (type EC_T_BYTE)

EC_T_DWORD **dwBitSizeMoOpDisplay**
[in] Bit size of mode of operation display in process data image

EC_T_DWORD **dwBitOffsDigitalInputs**
[in] Bit offset within input process data image for digital inputs of axis (type EC_T_DWORD)

EC_T_DWORD **dwBitSizeDigitalInputs**
[in] Bit size of digital inputs in process data image

struct **MC_T_AXIS_ECAT_PARMS**

Public Members

EC_T_DWORD **dwSlaveId**
[in] Identification of slave

EC_T_WORD **wStationAddress**
[in] Station address of slave

EC_T_WORD **wCoeIdxOpMode**
[in] Index for object ‘Mode of Operation object’ (DS402 only). Default: 0x6060

EC_T_WORD **wCoeSubIdxOpMode**
[in] Subindex for object ‘Mode of Operation object’ (DS402 only). Default: 0

MC_T_DS402_OPERATION_MODE **eOpMode**
[in] Operation mode for DS402 axis

enum **MC_T_DS402_OPERATION_MODE**
Values:

enumerator **DRV_MODE_OP_UNDEFINED**
Undefined operation mode

enumerator **DRV_MODE_OP_CSP**
Cyclic Synchronous Position Mode

enumerator **DRV_MODE_OP_CSV**
Cyclic Synchronous Velocity Mode

enumerator **DRV_MODE_OP_CST**
Cyclic Synchronous Torque Mode

Example

```
AxisRef* S_pAxis = EC_NULL;
EC_T_DWORD dwRetVal = EC_E_ERROR;
MC_T_AXIS_PARMS oAxisParms;
MC_T_AXIS_IO_ENDPOINT oAxisIo;
MC_T_AXIS_ECAT_PARMS oAxisEcatParms;
OsMemset(&oAxisParms, 0, sizeof(MC_T_AXIS_PARMS));
OsMemset(&oAxisIo, 0, sizeof(MC_T_AXIS_IO_ENDPOINT));
OsMemset(&oAxisEcatParms, 0, sizeof(MC_T_AXIS_ECAT_PARMS));

oAxisParms.dwCycleTime = 1000;
oAxisParms.dwIncPerMM = 10000;
oAxisParms.dwMaxMoveCmds = 8;

oAxisIo.dwBitOffsStatusWord = 0; /* Determine offset by emGetSlaveInpVarByObjectEx␣
↪for example */
oAxisIo.dwBitSizeStatusWord = 8 * sizeof(EC_T_WORD);
oAxisIo.dwBitOffsActualPosition = 16; /* Determine offset by emGetSlaveInpVarByObjectEx␣
↪for example */
oAxisIo.dwBitSizeActualPosition = 8 * sizeof(EC_T_SDWORD);
oAxisIo.dwBitOffsControlWord = 0; /* Determine offset by emGetSlaveOutpVarByObjectEx␣
↪for example */
oAxisIo.dwBitSizeControlWord = 8 * sizeof(EC_T_WORD);
oAxisIo.dwBitOffsTargetPosition = 16; /* Determine offset by emGetSlaveOutpVarByObjectEx␣
↪for example */
oAxisIo.dwBitSizeTargetPosition = 8 * sizeof(EC_T_SDWORD);

oAxisEcatParms.dwSlaveId = 1; /* Determine value by emGetCfgSlaveInfo for example␣
↪*/
oAxisEcatParms.wStationAddress = 1001;
oAxisEcatParms.wCoeIdxOpMode = 0x6060;
oAxisEcatParms.wCoeSubIdxOpMode = 0;
oAxisEcatParms.eOpMode = DRV_MODE_OP_CSP;

dwRetVal = mcCreateDS402Axis(oAxisParms, oAxisIo, oAxisEcatParms, S_pAxis);
```

3.3 Cyclic functions

The EC-Motion-Advanced library itself does not create any threads. The application must therefore cyclically call some functions to update the inputs and outputs and, for example, calculate new target positions. A description of these function is given in the next sections.

3.3.1 mcRefreshAxisInputs

```
EC_T_DWORD mcRefreshAxisInputs (
    EC_T_DWORD dwInstanceId,
    EC_T_DWORD dwTaskId,
    EC_T_BYTE *pbyInProcessImage
)
```

After receiving new inputs from EtherCAT frames call this function in order to update the axis state like status word or actual position according to given inputs. This is done only if the corresponding slave is in the state SAFEOP or OP.

Parameters

- **dwInstanceId** – Identification of instance.
- **dwTaskId** – Identification of task.
- **pbyInProcessImage** – Pointer to current input process image.

Returns

When axis inputs were updated successfully EC_E_NOERROR is returned. Otherwise, an error code is given indicating the cause of the failure.

3.3.2 mcRefreshAxisOutputs

```
EC_T_DWORD mcRefreshAxisOutputs (
    EC_T_DWORD dwInstanceId,
    EC_T_DWORD dwTaskId,
    EC_T_BYTE *pbyOutProcessImage
)
```

Call this function before new EtherCAT frames are sent to the subdevices in order to update the outputs like control word or target position in the EtherCAT output process image according to the current state of the axis.

Parameters

- **dwInstanceId** – Identification of instance.
- **dwTaskId** – Identification of task.
- **pbyOutProcessImage** – Pointer to current output process image.

Returns

When outputs were updated successfully EC_E_NOERROR is returned. Otherwise, an error code is given indicating the cause of the failure.

3.3.3 mcCyclicTask

EC_T_DWORD **mcCyclicTask** (EC_T_DWORD dwInstanceId, EC_T_DWORD dwTaskId)

Execute the cyclic tasks of axes belonging to an execution unit given by instance and task id. This includes the calculation of the next target position of the axes during a movement. During this function call no motion function block should be operated. This will yield to undefined behaviour.

Parameters

- **dwInstanceId** – Identification of instance.
- **dwTaskId** – Identification of task.

Returns

If cyclic tasks are performed successfully, EC_E_NOERROR is returned. Otherwise an error id is given identifying the error.

Example

```
EC_T_DWORD dwRes = EC_E_ERROR;
EC_T_BYTE* pbyPdIn = ecatGetProcessImageInputPtr();
EC_T_BYTE* pbyPdOut = ecatGetProcessImageOutputPtr();

dwRes = mcRefreshAxisInputs(INSTANCE_DEFAULT, 0, pbyPdIn);

/*
 * Implementation of the application.
 * E.g. call of motion function blocks McPower and McMoveRelative
 */

dwRes = mcCyclicTask(INSTANCE_DEFAULT, 0);

dwRes = mcRefreshAxisOutputs(INSTANCE_DEFAULT, 0, pbyPdOut);
```

3.4 General functions

3.4.1 mcGetText

const EC_T_CHAR ***mcGetText** (EC_T_DWORD dwTextId)

Return text tokens by ID.

Parameters

dwTextId – ID where text should be returned.

Returns

Textual description of the given ID.

3.5 Motion Control Function Blocks

Motion control function blocks are designed as callable C++ classes. They consist of a set of input variables which are readable and writable and a set of output variables which are only readable. Furthermore almost all function blocks have an instance of *AxisRef* as input output variable. In the description of the function blocks basic variables are marked with '**B**' and optional parameters are marked with '**E**'. Additional variables are indicated by '**V**'.

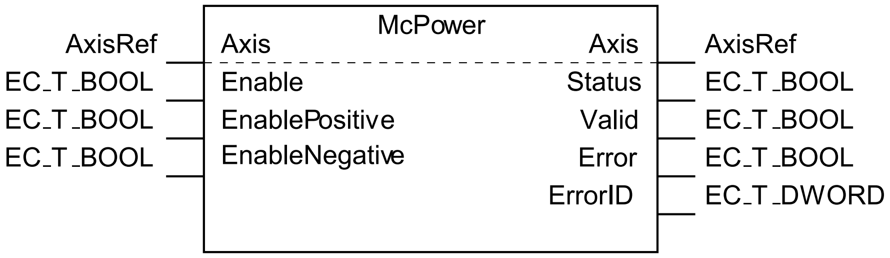
Each function block is callable. This means that an instance of the function block can be used as a function. This executes the function block in the sense that the desired operation is performed on the specified axis. The input variables have to be set before the call, the output variables are calculated during the call and could be read afterwards. Not all inputs need to be defined. In this case, the input values from the previous call are used, or default values are used when the function block is invoked for the first time.

Example

```
/* cyclic call within the application */
S_oMcPower.Axis = S_pAxis;
S_oMcPower.Enable = EC_TRUE;
S_oMcPower();
if (S_oMcPower.Error)
{
    EcLogMsg(EC_LOG_LEVEL_ERROR, (pEcLogContext, EC_LOG_LEVEL_ERROR,
        "Error 0x%x during power on of axis!", (EC_T_DWORD)S_oMcPower.ErrorID));
}
```

3.5.1 McPower

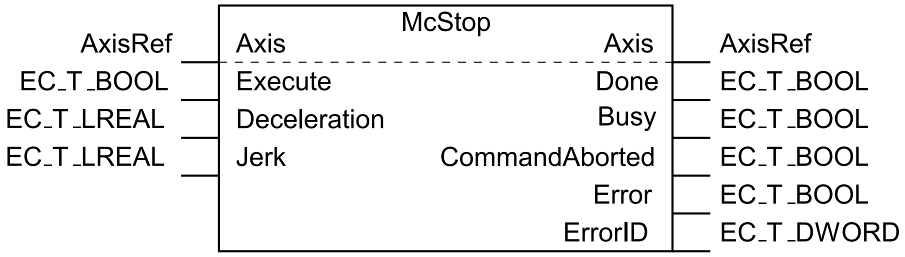
Function block			McPower	
This function block controls the power stage (On or Off)				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Enable	EC_T_BOOL	As long as 'Enable' is true, power is being enabled.
	E	EnablePositive	EC_T_BOOL	As long as 'Enable' is true, this permits motion in positive direction.
	E	EnableNegative	EC_T_BOOL	As long as 'Enable' is true, this permits motion in negative direction.
VAR_OUT				
	B	Status	EC_T_BOOL	Effective state of the power stage.
	E	Valid	EC_T_BOOL	If true, a valid set of outputs is available at the function block.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.



Hint: Per axis only one function block McPower should be issued. Otherwise an error is given.

3.5.2 McStop

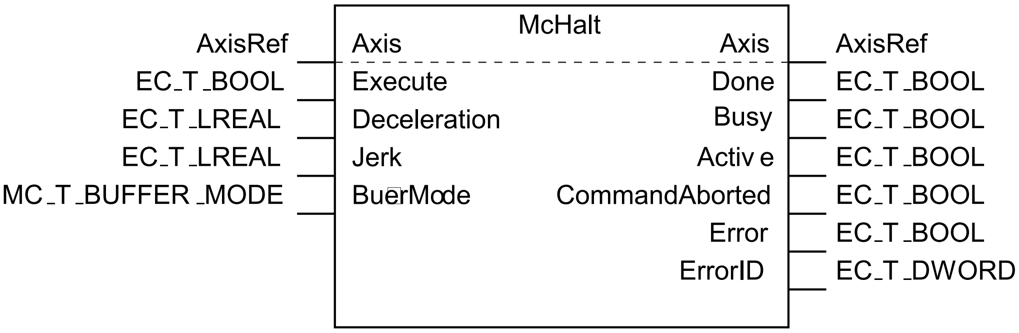
Function block			McStop	
This function block commands a controlled motion stop and transfers the axis to the state ‘Stopping’. It aborts any ongoing function block execution. While the axis is in state ‘Stopping’, no other function block can perform any motion on the same axis. After the axis has reached ‘Velocity’ zero, the ‘Done’ output is set to TRUE immediately. The axis remains in the state ‘Stopping’ as long as ‘Execute’ is still TRUE or ‘Velocity’ zero is not yet reached. As soon as ‘Done’ is SET and ‘Execute’ is FALSE the axis goes to state ‘Standstill’.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Execute	EC_T_BOOL	Start the action at rising edge.
	E	Deceleration	EC_T_LREAL	Value of the deceleration [u/s^2].
	E	Jerk	EC_T_LREAL	Value of the jerk [u/s^3].
VAR_OUT				
	B	Done	EC_T_BOOL	Zero velocity reached.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	E	CommandAborted	EC_T_BOOL	‘Command’ is aborted by switching off power (only possibility to abort).
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.



Hint: As long as ‘Execute’ is high, the axis remains in the state ‘Stopping’ and may not be executing any other motion command.

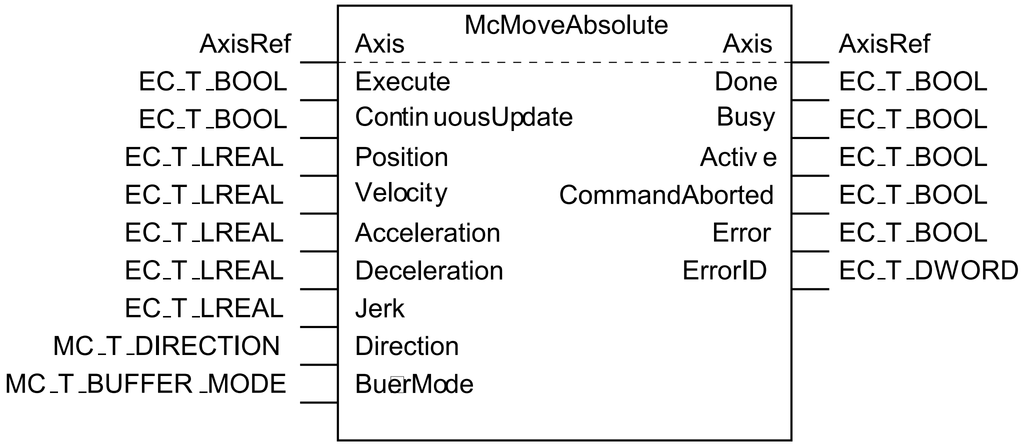
3.5.3 McHalt

Function block			McHalt	
This function block commands a controlled motion stop. The axis is moved to the state ‘DiscreteMotion’, until the velocity is zero. With the ‘Done’ output set, the state is transferred to ‘Standstill’.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Execute	EC_T_BOOL	Start the action at rising edge.
	E	Deceleration	EC_T_LREAL	Value of the deceleration [u/s^2].
	E	Jerk	EC_T_LREAL	Value of the jerk [u/s^3].
	E	BufferMode	MC_T_BUFFER_MODE	Defines the chronological sequence of the function block.
VAR_OUT				
	B	Done	EC_T_BOOL	Zero velocity reached.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	E	Active	EC_T_BOOL	Indicates that the function block has control on the axis.
	E	CommandAborted	EC_T_BOOL	‘Command’ is aborted by another command.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.



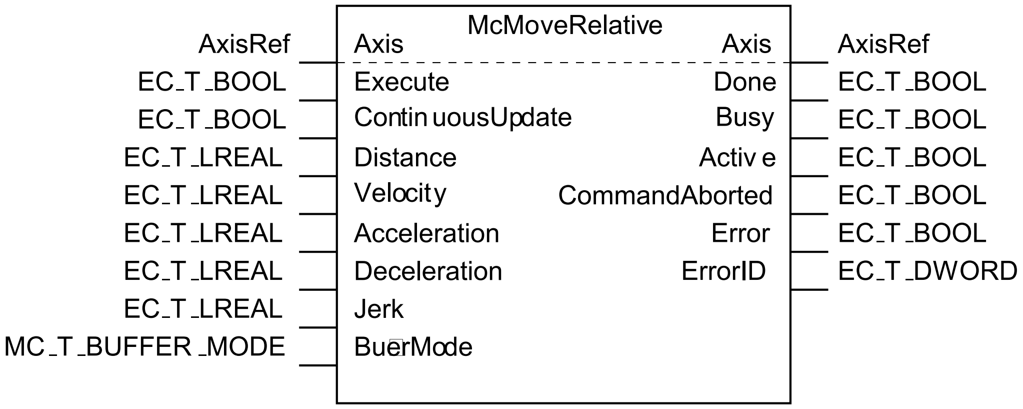
3.5.4 McMoveAbsolute

Function block			McMoveAbsolute	
This function block commands a controlled motion to a specified absolute position.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Execute	EC_T_BOOL	Start the motion at rising edge.
	E	ContinuousUpdate	EC_T_BOOL	If TRUE use the current values of the input variables and apply it to the ongoing movement.
	B	Position	EC_T_LREAL	Commanded position for the motion (in technical unit [u]).
	B	Velocity	EC_T_LREAL	Value of the maximum velocity (not necessarily reached) [u/s].
	E	Acceleration	EC_T_LREAL	Value of the acceleration (increasing energy of the motor) [u/s^2].
	E	Deceleration	EC_T_LREAL	Value of the deceleration (decreasing energy of the motor) [u/s^2].
	E	Jerk	EC_T_LREAL	Value of the Jerk [u/s^3].
	B	Direction	MC_T_DIRECTION	Determine the approach to calculate the path to move for a modulo axis.
	E	BufferMode	MC_T_BUFFER_MODE	Defines the chronological sequence of the function block.
VAR_OUT				
	B	Done	EC_T_BOOL	Commanded distance reached.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	E	Active	EC_T_BOOL	Indicates that the function block has control on the axis.
	E	CommandAborted	EC_T_BOOL	‘Command’ is aborted by another command.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.



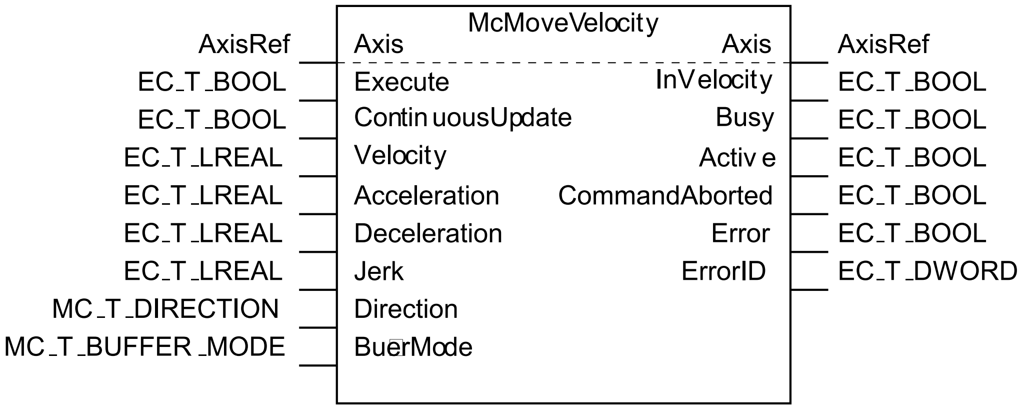
3.5.5 McMoveRelative

Function block			McMoveRelative	
This function block commands a controlled motion of a specific distance relative to the set position at the time of the execution.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Execute	EC_T_BOOL	Start the motion at rising edge.
	E	ContinuousUpdate	EC_T_BOOL	If TRUE use the current values of the input variables and apply it to the ongoing movement.
	B	Distance	EC_T_LREAL	Relative distance for the motion (in technical unit [u]).
	B	Velocity	EC_T_LREAL	Value of the maximum velocity (not necessarily reached) [u/s].
	E	Acceleration	EC_T_LREAL	Value of the acceleration (increasing energy of the motor) [u/s^2].
	E	Deceleration	EC_T_LREAL	Value of the deceleration (decreasing energy of the motor) [u/s^2].
	E	Jerk	EC_T_LREAL	Value of the Jerk [u/s^3].
	E	BufferMode	MC_T_BUFFER_MODE	Defines the chronological sequence of the function block.
VAR_OUT				
	B	Done	EC_T_BOOL	Commanded distance reached.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	E	Active	EC_T_BOOL	Indicates that the function block has control on the axis.
	E	CommandAborted	EC_T_BOOL	'Command' is aborted by another command.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.



3.5.6 McMoveVelocity

Function block			McMoveVelocity	
This function block commands a never ending controlled motion at a specified velocity.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Execute	EC_T_BOOL	Start the motion at rising edge.
	E	ContinuousUpdate	EC_T_BOOL	If TRUE use the current values of the input variables and apply it to the ongoing movement.
	B	Velocity	EC_T_LREAL	Value of the maximum velocity [u/s]. Can be a signed value.
	E	Acceleration	EC_T_LREAL	Value of the acceleration (increasing energy of the motor) [u/s^2].
	E	Deceleration	EC_T_LREAL	Value of the deceleration (decreasing energy of the motor) [u/s^2].
	E	Jerk	EC_T_LREAL	Value of the Jerk [u/s^3].
	E	Direction	MC_T_DIRECTION	Direction of movement.
	E	BufferMode	MC_T_BUFFER_MODE	Defines the chronological sequence of the function block.
VAR_OUT				
	B	InVelocity	EC_T_BOOL	Commanded velocity reached.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	E	Active	EC_T_BOOL	Indicates that the function block has control on the axis.
	E	CommandAborted	EC_T_BOOL	‘Command’ is aborted by another command.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.

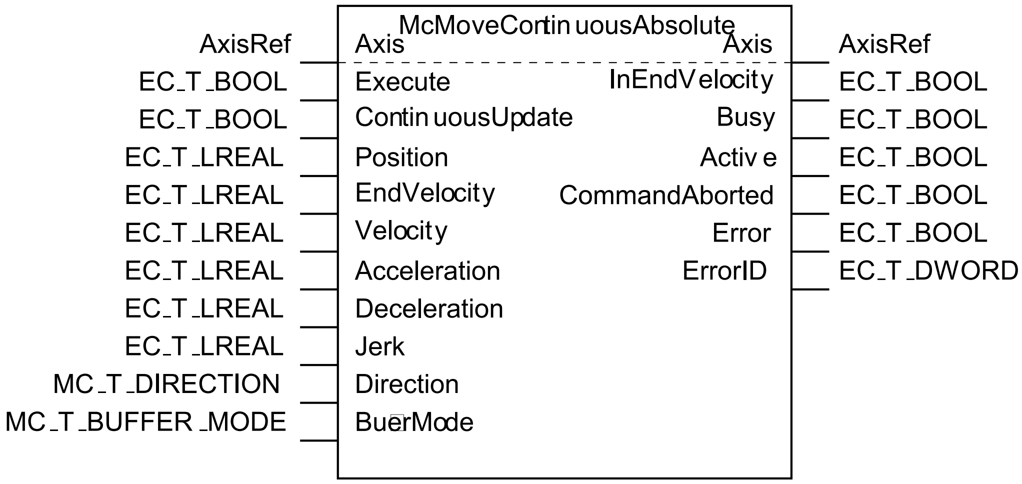


Hint: In order to stop the motion, the function block has to be interrupted by another function block issuing a new motion command.

Hint: A negative velocity combined with a negative direction leads to a positive velocity.

3.5.7 McMoveContinuousAbsolute

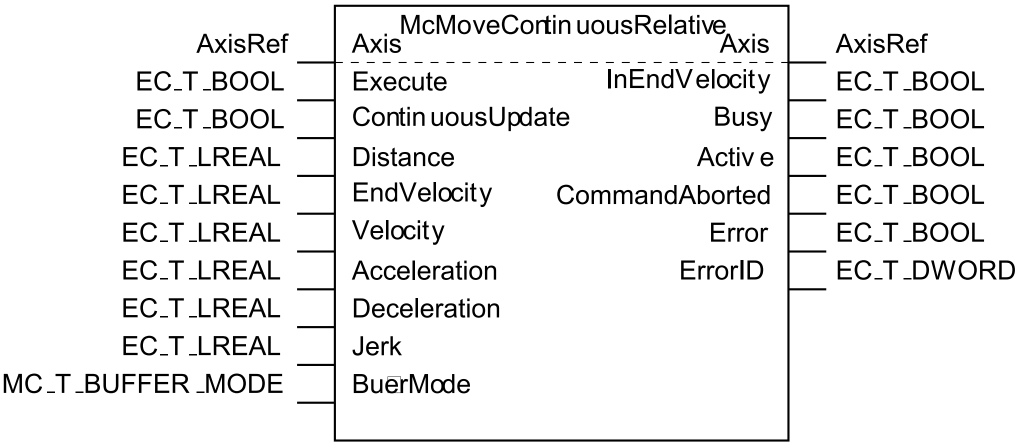
Function block		McMoveContinuousAbsolute	
This function block commands a controlled motion to a specified absolute position ending with the specified velocity.			
VAR_INOUT			
B	Axis	AxisRef	Reference to the axis
VAR_IN			
B	Execute	EC_T_BOOL	Start the motion at rising edge.
E	ContinuousUpdate	EC_T_BOOL	If TRUE use the current values of the input variables and apply it to the ongoing movement.
B	Position	EC_T_LREAL	Commanded position for the motion (in technical unit [u]).
B	EndVelocity	EC_T_LREAL	Value of the end velocity [u/s]. Signed value.
B	Velocity	EC_T_LREAL	Value of the maximum velocity (not necessarily reached) [u/s].
E	Acceleration	EC_T_LREAL	Value of the acceleration (increasing energy of the motor) [u/s^2].
E	Deceleration	EC_T_LREAL	Value of the deceleration (decreasing energy of the motor) [u/s^2].
E	Jerk	EC_T_LREAL	Value of the Jerk [u/s^3].
B	Direction	MC_T_DIRECTION	Determine the approach to calculate the path to move for a modulo axis.
E	BufferMode	MC_T_BUFFER_MODE	Defines the chronological sequence of the function block.
VAR_OUT			
B	InEndVelocity	EC_T_BOOL	Commanded position reached and running at requested end velocity.
E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
E	Active	EC_T_BOOL	Indicates that the function block has control on the axis.
E	CommandAborted	EC_T_BOOL	‘Command’ is aborted by another command.
B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
E	ErrorID	EC_T_DWORD	Error identification.



Hint: If the commanded position is reached and no new motion command is given, the axis continues to run with the specified ‘EndVelocity’.

3.5.8 McMoveContinuousRelative

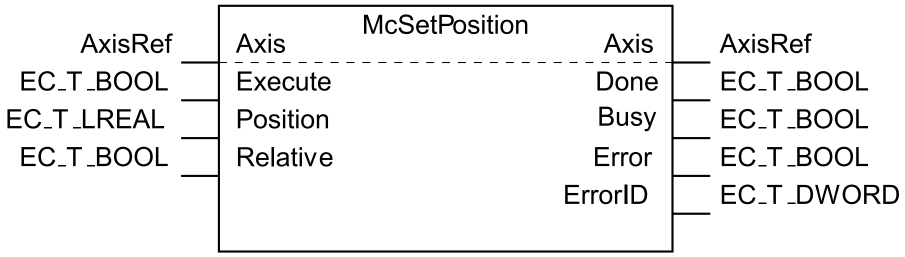
Function block			McMoveContinuousRelative	
This function block commands a controlled motion of a specific relative distance ending with the specified velocity.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Execute	EC_T_BOOL	Start the motion at rising edge.
	E	ContinuousUpdate	EC_T_BOOL	If TRUE use the current values of the input variables and apply it to the ongoing movement.
	B	Distance	EC_T_LREAL	Relative distance for the motion (in technical unit [u]).
	B	EndVelocity	EC_T_LREAL	Value of the end velocity [u/s]. Signed value.
	B	Velocity	EC_T_LREAL	Value of the maximum velocity (not necessarily reached) [u/s].
	E	Acceleration	EC_T_LREAL	Value of the acceleration (increasing energy of the motor) [u/s^2].
	E	Deceleration	EC_T_LREAL	Value of the deceleration (decreasing energy of the motor) [u/s^2].
	E	Jerk	EC_T_LREAL	Value of the Jerk [u/s^3].
	E	BufferMode	MC_T_BUFFER_MODE	Defines the chronological sequence of the function block.
VAR_OUT				
	B	InEndVelocity	EC_T_BOOL	Commanded distance reached and running at requested end velocity.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	E	Active	EC_T_BOOL	Indicates that the function block has control on the axis.
	E	CommandAborted	EC_T_BOOL	'Command' is aborted by another command.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.



Hint: If the commanded position is reached and no new motion command is given, the axis continues to run with the specified 'EndVelocity'.

3.5.9 McSetPosition

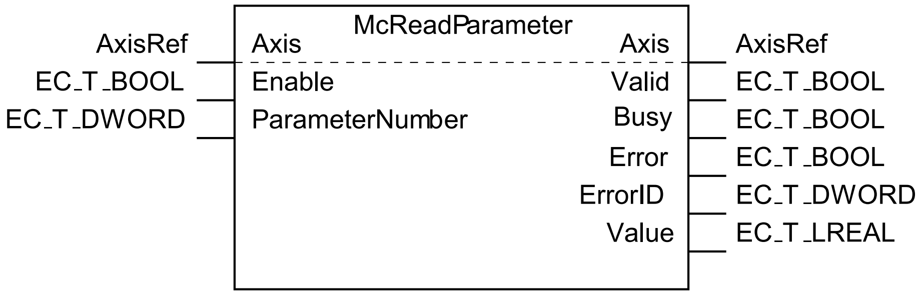
Function block			McSetPosition	
This function block shifts the coordinate system of an axis by manipulating both the set-point position as well as the actual position of an axis with the same value without any movement caused. This can be used for instance for a reference situation. This function block can also be used during motion without changing the commanded position, which is now positioned in the shifted coordinate system.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Execute	EC_T_BOOL	Start setting position in axis.
	B	Position	EC_T_LREAL	Position in unit [u] (Means 'Distance' if 'Relative' is TRUE).
	E	Relative	EC_T_BOOL	'Relative' distance if TRUE, 'Absolute' position if FALSE (default).
VAR_OUT				
	B	Done	EC_T_BOOL	Position has new value.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.



Hint: 'Relative' means that 'Position' is added to the actual position value of the axis at the time of execution. This results in a recalibration by a specified distance. 'Absolute' means that the actual position value of the axis is set to the value of specified in the 'Position' parameter.

3.5.10 McReadParameter

Function block		McReadParameter		
This function block returns the value of a specific parameter. The returned value is converted to Real if necessary.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Enable	EC_T_BOOL	Get the value of the parameter continuously while enabled.
	B	Parameternumber	EC_T_DWORD	Number of the parameter.
VAR_OUT				
	B	Valid	EC_T_BOOL	A valid output is available at the function block.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.
	B	Value	EC_T_LREAL	Value of the specified parameter.



A list of supported parameter number is given by the enumeration *MC_T_PARAMETER_NUMBER*. It contains both readable and writable parameter.

enum **MC_T_PARAMETER_NUMBER**
Values:

enumerator **MC_PN_COMMANDED_POSITION**
REAL: Commanded position (Read)

enumerator **MC_PN_SOFTWARE_LIMIT_POS**
REAL: Positive software limit switch position (Read/Write)

enumerator **MC_PN_SOFTWARE_LIMIT_NEG**
REAL: Negative software limit switch position (Read/Write)

enumerator **MC_PN_ENABLE_LIMIT_POS**
BOOL: Enable positive software limit switch (Read/Write)

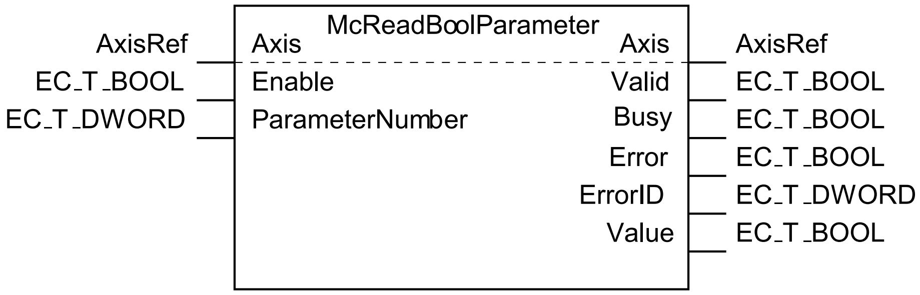
enumerator **MC_PN_ENABLE_LIMIT_NEG**
BOOL: Enable negative software limit switch (Read/Write)

enumerator **MC_PN_ACTUAL_VELOCITY**
REAL: Actual velocity (Read)

enumerator **MC_PN_COMMANDED_VELOCITY**
REAL: Commanded velocity (Read)

3.5.11 McReadBoolParameter

Function block			McReadBoolParameter	
This function block returns the value of a specific parameter with data type BOOL.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Enable	EC_T_BOOL	Get the value of the parameter continuously while enabled.
	B	Parameternumber	EC_T_DWORD	Number of the parameter.
VAR_OUT				
	B	Valid	EC_T_BOOL	A valid output is available at the function block.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.
	B	Value	EC_T_BOOL	Value of the specified parameter.

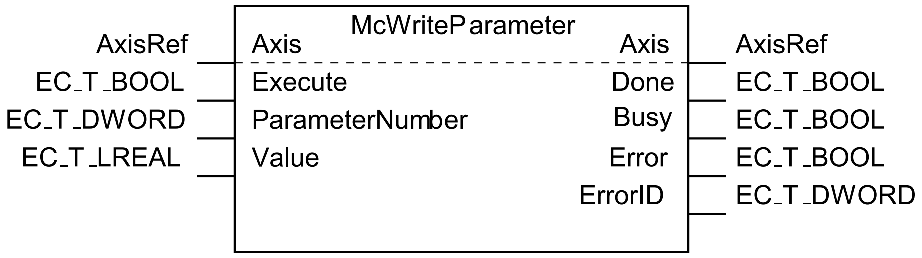


See also:

MC_T_PARAMETER_NUMBER

3.5.12 McWriteParameter

Function block			McWriteParameter	
This function block modifies the value of a specific parameter.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Execute	EC_T_BOOL	Write the value of the parameter at rising edge.
	B	Parameternumber	EC_T_DWORD	Number of the parameter.
	B	Value	EC_T_LREAL	New value of the specified parameter.
VAR_OUT				
	B	Done	EC_T_BOOL	Parameter successfully written.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.

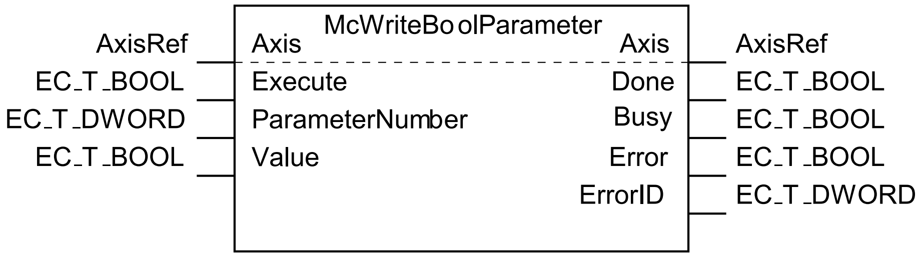


See also:

MC_T_PARAMETER_NUMBER

3.5.13 McWriteBoolParameter

Function block			McWriteBoolParameter	
This function block modifies the value of a specific parameter of data type BOOL.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Execute	EC_T_BOOL	Write the value of the parameter at rising edge.
	B	Parameternumber	EC_T_DWORD	Number of the parameter.
	B	Value	EC_T_BOOL	New value of the specified parameter.
VAR_OUT				
	B	Done	EC_T_BOOL	Parameter successfully written.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.

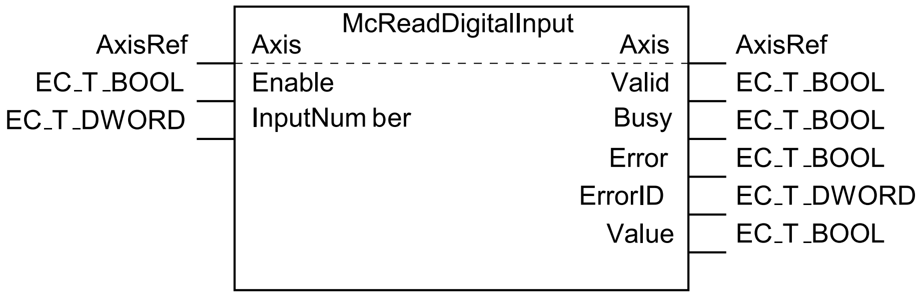


See also:

MC_T_PARAMETER_NUMBER

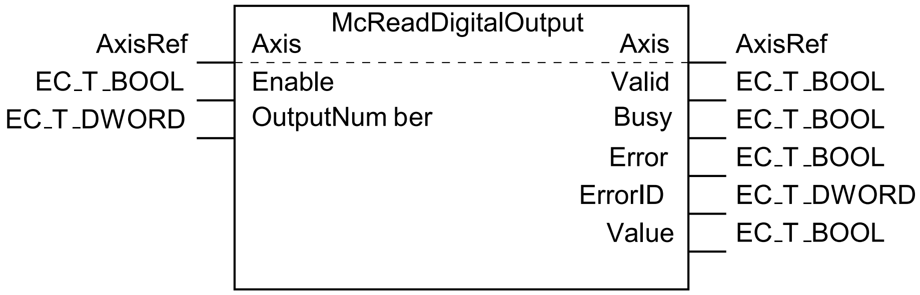
3.5.14 McReadDigitalInput

Function block			McReadDigitalInput	
This function block gives access to the value of the digital input of an axis. It provides the value of the referenced input.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Enable	EC_T_BOOL	Get the value of the selected input signal continuously while enabled.
	E	Inputnumber	EC_T_DWORD	Selects the digital input.
VAR_OUT				
	B	Valid	EC_T_BOOL	A valid output is available at the function block.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.
	B	Value	EC_T_BOOL	The value of the selected input signal.



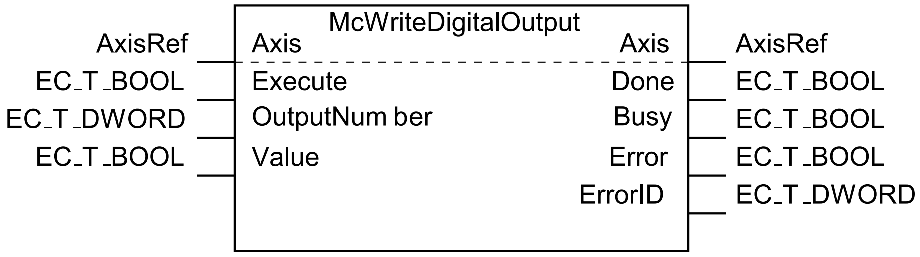
3.5.15 McReadDigitalOutput

Function block			McReadDigitalOutput	
This function block gives access to the value of the digital output of an axis. It provides the value of the referenced output.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Enable	EC_T_BOOL	Get the value of the selected output signal continuously while enabled.
	E	Outputnumber	EC_T_DWORD	Selects the digital output.
VAR_OUT				
	B	Valid	EC_T_BOOL	A valid output is available at the function block.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.
	B	Value	EC_T_BOOL	The value of the selected output signal.



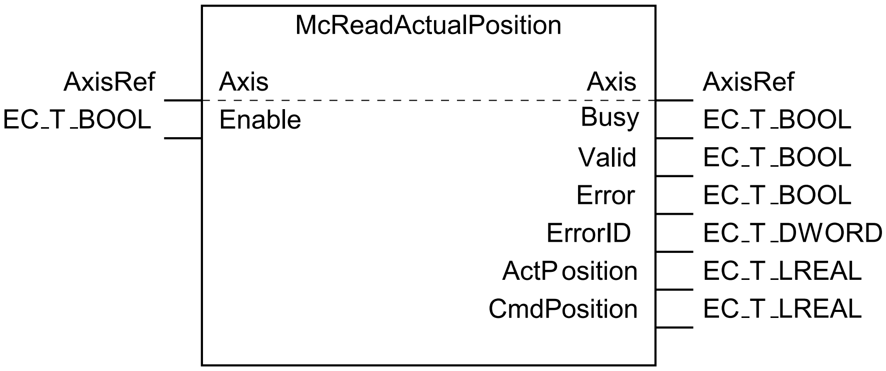
3.5.16 McWriteDigitalOutput

Function block			McWriteDigitalOutput	
This function block write a value to the digital output once.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Execute	EC_T_BOOL	Write the value of the selected digital output at rising edge.
	E	Outputnumber	EC_T_DWORD	Selects the digital output.
	B	Value	EC_T_BOOL	The value of the selected digital output.
VAR_OUT				
	B	Done	EC_T_BOOL	Writing of the digital output signal is done.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.



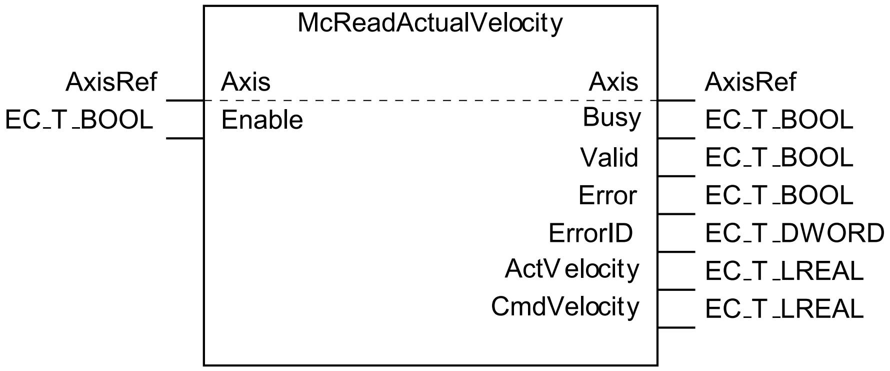
3.5.17 McReadActualPosition

Function block			McReadActualPosition	
This function block returns the actual position.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Enable	EC_T_BOOL	Get the value of the parameter continuously while enabled.
VAR_OUT				
	B	Valid	EC_T_BOOL	A valid output is available at the function block.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.
	B	ActPosition	EC_T_LREAL	New absolute position (in axis' unit [u]).
	V	CmdPosition	EC_T_LREAL	Commanded position (in axis' unit [u]).



3.5.18 McReadActualVelocity

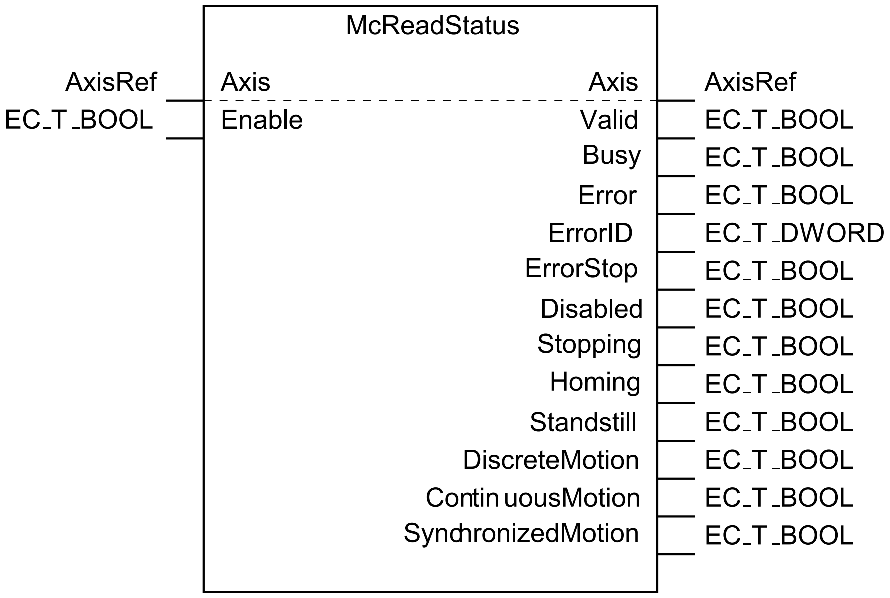
Function block			McReadActualVelocity	
This function block returns the value of the actual velocity as long as 'Enable' is set. 'Valid' is true when the data output 'Velocity' is valid. If 'Enable' is reset, the data loses its validity, and all outputs are reset, no matter if new data is available.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Enable	EC_T_BOOL	Get the value of the parameter continuously while enabled.
VAR_OUT				
	B	Valid	EC_T_BOOL	A valid output is available at the function block.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.
	B	ActVelocity	EC_T_LREAL	The value of the actual velocity (in axis' unit [u/s]) .
	V	CmdVelocity	EC_T_LREAL	The value of the commanded velocity (in axis' unit [u/s]) .



Hint: The output ‘Velocity’ is a signed value.

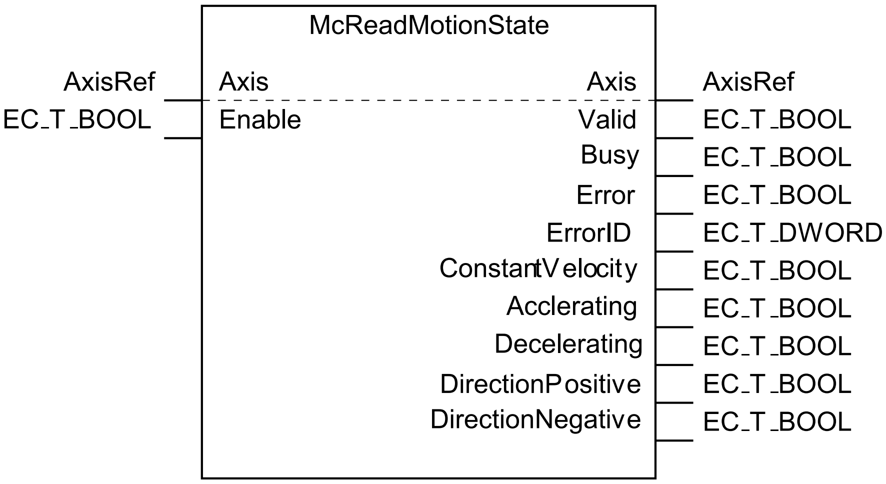
3.5.19 McReadStatus

Function block			McReadStatus	
This function block returns in detail the status of the state diagram of the selected axis.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Enable	EC_T_BOOL	Get the value of the parameter continuously while enabled.
VAR_OUT				
	B	Valid	EC_T_BOOL	A valid set of outputs is available at the function block.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.
	B	ErrorStop	EC_T_BOOL	Axis is at state ERRORSTOP.
	B	Disabled	EC_T_BOOL	Axis is at state DISABLED.
	B	Stopping	EC_T_BOOL	Axis is at state STOPPING.
	E	Homing	EC_T_BOOL	Axis is at state HOMING.
	B	Standstill	EC_T_BOOL	Axis is at state STANDSTILL.
	E	DiscreteMotion	EC_T_BOOL	Axis is at state DISCRETEMOTION.
	E	ContinuousMotion	EC_T_BOOL	Axis is at state CONTINUOUSMOTION.
	E	SynchronizedMotion	EC_T_BOOL	Axis is at state SYNCHRONIZEDMOTION.



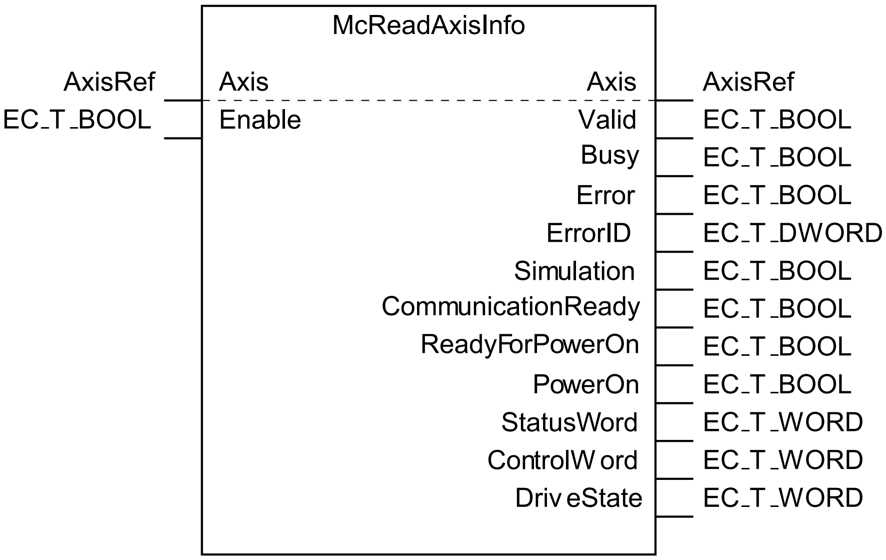
3.5.20 McReadMotionState

Function block			McReadMotionState	
This function block returns in detail the status of the axis with respect to the motion currently in progress.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Enable	EC_T_BOOL	Get the value of the parameter continuously while enabled.
VAR_OUT				
	B	Valid	EC_T_BOOL	A valid set of outputs is available at the function block.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.
	E	ConstantVelocity	EC_T_BOOL	Velocity is constant. Velocity may be zero.
	E	Accelerating	EC_T_BOOL	Increasing the absolute value of the velocity.
	E	Decelerating	EC_T_BOOL	Decreasing the absolute value of the velocity.
	E	DirectionPositive	EC_T_BOOL	Signals that the position is increasing.
	E	DirectionNegative	EC_T_BOOL	Signals that the position is decreasing.



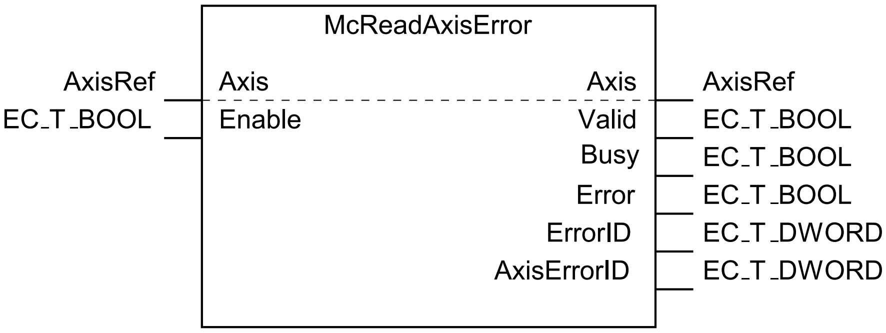
3.5.21 McReadAxisInfo

Function block		McReadAxisInfo	
This function block reads information concerning an axis, like modes, inputs directly related to the axis, and certain status information.			
VAR_INOUT			
B	Axis	AxisRef	Reference to the axis
VAR_IN			
B	Enable	EC_T_BOOL	Get the axis information constantly while enabled.
VAR_OUT			
B	Valid	EC_T_BOOL	True if a valid set of outputs is available.
E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
E	ErrorID	EC_T_DWORD	Error identification.
E	Simulation	EC_T_BOOL	Axis is in simulation mode.
E	Communication-Ready	EC_T_BOOL	‘Network’ is initialized and ready for communication.
E	ReadyForPowerOn	EC_T_BOOL	Drive is ready to be enabled (power on).
E	PowerOn	EC_T_BOOL	If true shows that the power stage is switched on.
V	StatusWord	EC_T_WORD	Status word of axis.
V	ControlWord	EC_T_WORD	Control word of axis.
V	DriveState	EC_T_WORD	Drive state of axis.



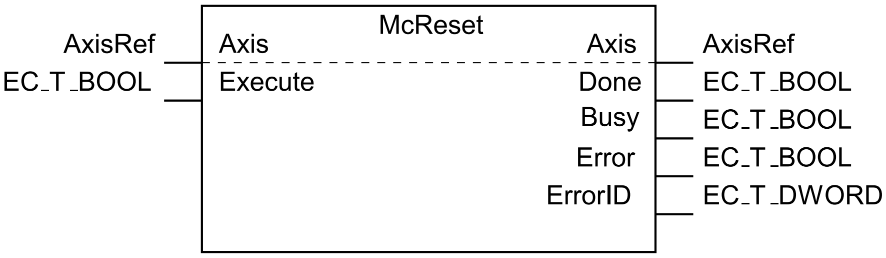
3.5.22 McReadAxisError

Function block			McReadAxisError	
This function block presents general axis errors not relating to the function blocks (for instance axis errors, drive errors, communication errors).				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the axis
VAR_IN				
	B	Enable	EC_T_BOOL	Get the value of the parameter continuously while enabled.
VAR_OUT				
	B	Valid	EC_T_BOOL	True if a valid output is available at the function block.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.
	E	AxisErrorID	EC_T_DWORD	The value of the axis error.



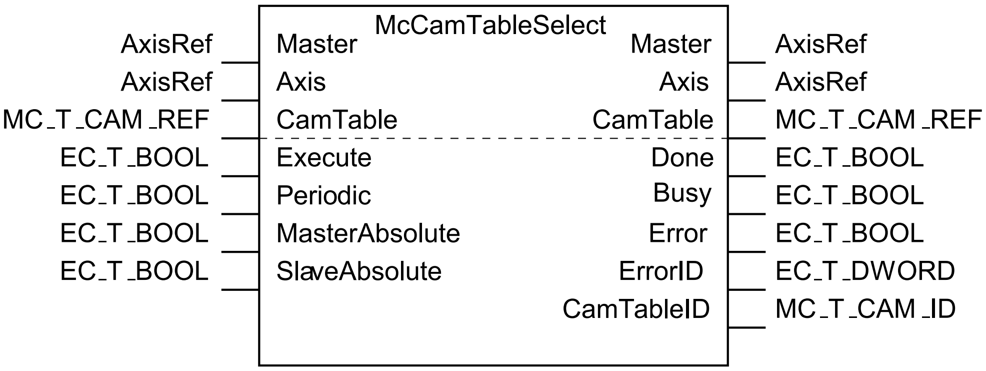
3.5.23 McReset

Function block		McReset	
This function block makes the transition from the state ‘ErrorStop’ to ‘Standstill’ or ‘Disabled’ by resetting all internal axis-related errors - it does not affect the output of the function block instances.			
VAR_INOUT			
B	Axis	AxisRef	Reference to the axis
VAR_IN			
B	Execute	EC_T_BOOL	Resets all internal axis-related errors.
VAR_OUT			
B	Done	EC_T_BOOL	‘Standstill’ or ‘Disabled’ state is reached.
E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
E	ErrorID	EC_T_DWORD	Error identification.



3.5.24 McCamTableSelect

Function block		McCamTableSelect	
This function block selects the CAM tables by setting the connections to the relevant tables.			
VAR_INOUT			
E	Master	AxisRef	Reference to the master axis.
E	Axis	AxisRef	Reference to the slave axis.
B	CamTable	MC_T_CAM_REF	Reference to CAM description.
VAR_IN			
B	Execute	EC_T_BOOL	Selection at rising edge.
E	Periodic	EC_T_BOOL	Distinguish between periodic and non-periodic (single shot) mode.
E	MasterAbsolute	EC_T_BOOL	Distinguish between absolute and relative coordinates of master axis.
E	SlaveAbsolute	EC_T_BOOL	Distinguish between absolute and relative coordinates of slave axis.
VAR_OUT			
B	Done	EC_T_BOOL	Pre-selection done.
E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
E	ErrorID	EC_T_DWORD	Error identification.
E	CamTableID	MC_T_CAM_ID	Identifier of CAM table to be used in function block McCamIn.



struct **MC_T_CAM_REF**

Public Members

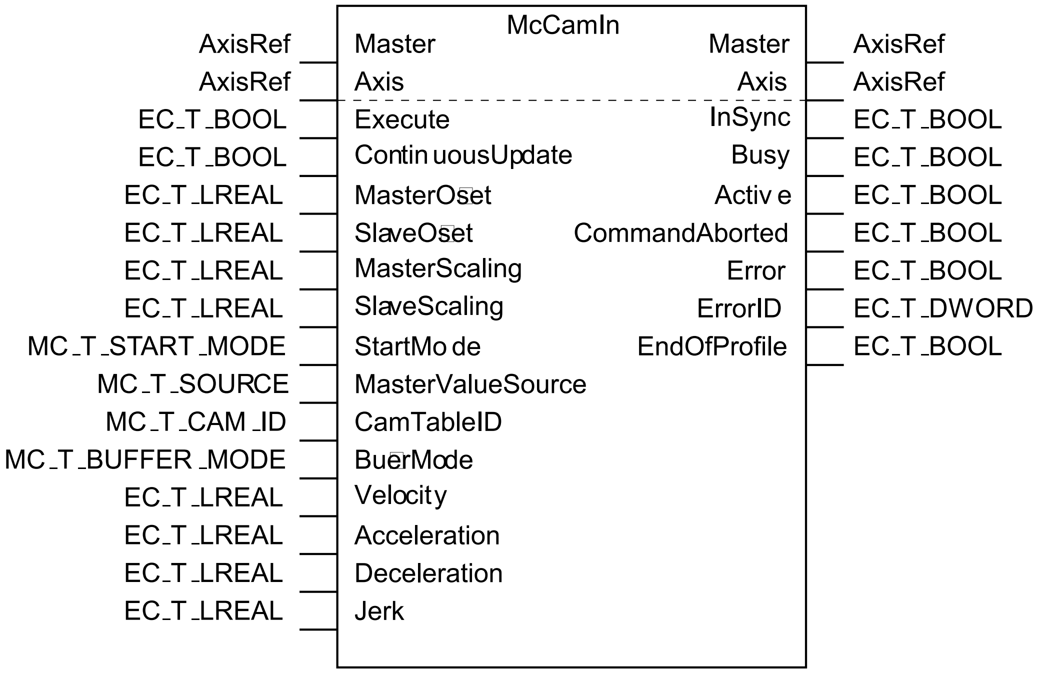
EC_T_DWORD **dwDegree**
[in] Polynomial degree for interpolation between knots.

EC_T_DWORD **dwNumOfElements**
[in] Number of elements within given data field.

EC_T_LREAL (***palfData**)[2]
[in] Two dimensional table with master/slave positions

3.5.25 McCamIn

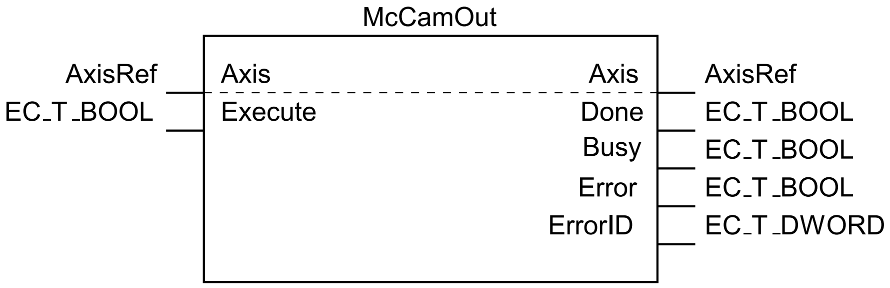
Function block			McCamIn	
This function block engages the CAM.				
VAR_INOUT				
	B	Master	AxisRef	Reference to the master axis.
	B	Axis	AxisRef	Reference to the slave axis.
VAR_IN				
	B	Execute	EC_T_BOOL	Start the CAM process at the rising edge.
	E	ContinuousUpdate	EC_T_BOOL	If TRUE use the current values of the input variables and apply it to the ongoing movement.
	E	MasterOffset	EC_T_LREAL	Offset of the master shaft to cam.
	E	SlaveOffset	EC_T_LREAL	Offset of slave table.
	E	MasterScaling	EC_T_LREAL	Factor for the master profile. From the slave point of view the master overall profile is multiplied by this factor.
	E	SlaveScaling	EC_T_LREAL	Factor for the slave profile. The overall slave profile is multiplied by this factor.
	E	StartMode	MC_T_START_MODE	Defines the mode how cam is started.
	E	MasterValueSource	MC_SOURCE	Defines the source for synchronization.
	E	CamTableID	MC_T_CAM_ID	Identifier of CAM table to be used, linked to the output of McCamTableSelect.
	E	BufferMode	MC_T_BUFFER_MODE	Defines the chronological sequence of the function block.
	V	Velocity	EC_T_LREAL	Value of the velocity for start mode RAMP_IN.
	V	Acceleration	EC_T_LREAL	Value of the acceleration for the start mode RAMP_IN.
	V	Deceleration	EC_T_LREAL	Value of the deceleration for the start mode RAMP_IN.
	V	Jerk	EC_T_LREAL	Value of the jerk for the start mode RAMP_IN.
VAR_OUT				
	B	InSync	EC_T_BOOL	The commanded value is equal to set value.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	E	Active	EC_T_BOOL	Indicates that the function block has control on the axis.
	E	CommandAborted	EC_T_BOOL	'Command' is aborted by another command.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.
	E	EndOfProfile	EC_T_BOOL	Pulsed output signaling the cyclic end of the CAM profile. It is displayed every time the end of the CAM profile is reached.



Hint: If the position of the slave axis does not correspond to the CAM profile, a jump at the slave axis will possibly occur.

3.5.26 McCamOut

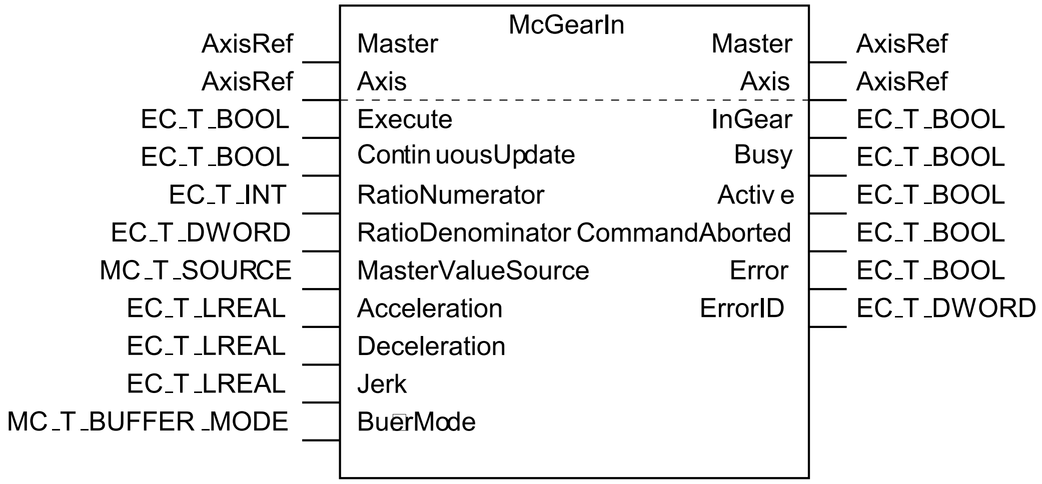
Function block			McCamOut	
This function block disengages the slave axis from the master axis immediately.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the slave axis.
VAR_IN				
	B	Execute	EC_T_BOOL	Start to disengage the slave from the master.
VAR_OUT				
	B	Done	EC_T_BOOL	Disengaging completed.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.



Hint: This command should be followed by another command, for instance McStop, McGearIn or any other move command. Otherwise the last velocity is maintained and there is no function block active on the slave axis till the next function block is issued.

3.5.27 McGearIn

Function block			McGearIn	
This function block commands a ratio between the velocity of the slave and master axis.				
VAR_INOUT				
	B	Master	AxisRef	Reference to the master axis.
	B	Axis	AxisRef	Reference to the slave axis.
VAR_IN				
	B	Execute	EC_T_BOOL	Start the gearing process at the rising edge.
	E	ContinuousUpdate	EC_T_BOOL	If TRUE use the current values of the input variables and apply it to the ongoing movement.
	B	RatioNumerator	EC_T_INT	Gear ratio numerator.
	B	RatioDenominator	EC_T_DWORD	Gear ratio denominator.
	E	MasterValueSource	MC_SOURCE	Defines the source for synchronization.
	E	Acceleration	EC_T_LREAL	Acceleration for gearing in.
	E	Deceleration	EC_T_LREAL	Deceleration for gearing in.
	E	Jerk	EC_T_LREAL	Jerk for gearing.
	E	BufferMode	MC_T_BUFFER_MODE	Defines the chronological sequence of the function block.
VAR_OUT				
	B	InGear	EC_T_BOOL	The commanded value is equal to set value.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	E	Active	EC_T_BOOL	Indicates that the function block has control on the axis.
	E	CommandAborted	EC_T_BOOL	‘Command’ is aborted by another command.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.

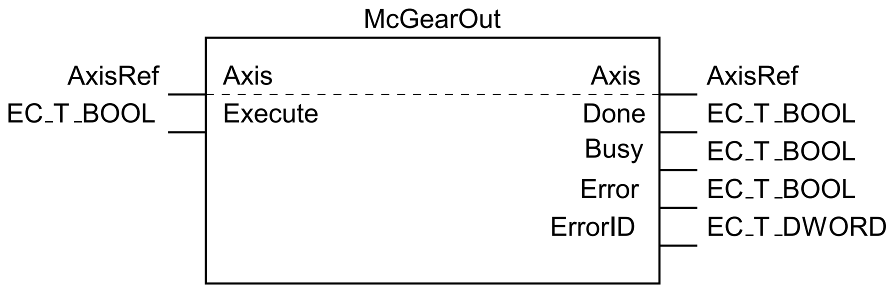


Hint: The slave axis ramps to the ratio of the master axis velocity and locks in when this is reached. Any lost distance during synchronization is not caught up.

Hint: Changing the gear ratio while McGearIn is running a consecutive McGearIn command can be used without the necessity to MCGearOut first.

3.5.28 McGearOut

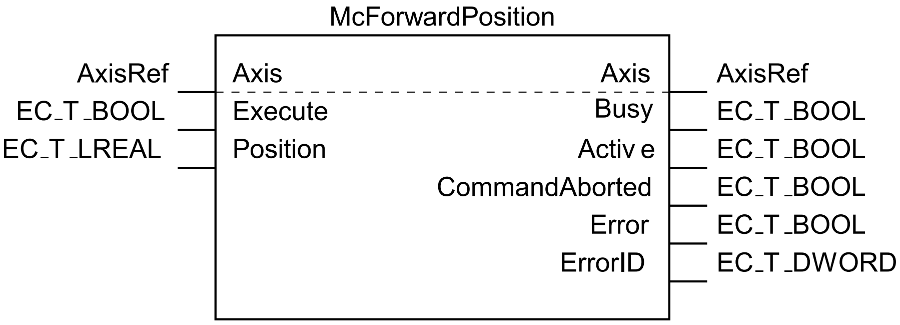
Function block			McGearOut	
This function block disengages the slave axis from the master axis.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to the slave axis.
VAR_IN				
	B	Execute	EC_T_BOOL	Start disengaging process at the rising edge.
VAR_OUT				
	B	Done	EC_T_BOOL	Disengaging completed.
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.



Hint: This command should be followed by another command, for instance McStop, McGearIn or any other move command. Otherwise the last velocity is maintained and there is no function block active on the slave axis till the next function block is issued.

3.5.29 McForwardPosition

Function block			McForwardPosition	
This function block sets the axis to the state SynchronizedMotion and forwards the position input directly to the commanded position.				
VAR_INOUT				
	B	Axis	AxisRef	Reference to axis.
VAR_IN				
	B	Execute	EC_T_BOOL	Start forwarding position at rising edge.
	B	Position	EC_T_LREAL	Position to forward.
VAR_OUT				
	E	Busy	EC_T_BOOL	The function block is not finished and new output values are to be expected.
	E	Active	EC_T_BOOL	Indicates that the function block has control on the axis.
	E	CommandAborted	EC_T_BOOL	‘Command’ is aborted by another command.
	B	Error	EC_T_BOOL	Signals that an error has occurred within the function block.
	E	ErrorID	EC_T_DWORD	Error identification.


McForwardPosition
AxisRef
EC_T_BOOL
EC_T_LREAL
Axis
Execute
Position
Axis
Busy
Activ e
CommandAborted
Error
ErrorID
AxisRef
EC_T_BOOL
EC_T_BOOL
EC_T_BOOL
EC_T_BOOL
EC_T_DWORD

4 Abbreviations

CAN	Controller Area Network
CANopen®	CiA's CAN application layer protocol
CiA	CAN in Automation
CiA 402	CANopen device profile for drives and motion control
CoE	CANopen over EtherCAT
CSP	Cyclic Synchronous Position mode (Operation mode of CiA 402 drives)
CST	Cyclic Synchronous Torque mode (Operation mode of CiA 402 drives)
CSV	Cyclic Synchronous Velocity mode (Operation mode of CiA 402 drives)
Drive	EtherCAT connected servo drive controller
DS402	Synonym for CiA 402
ETG	EtherCAT Technology Group
IEC	International Electrotechnical Commission
MCFB	Motion Control Function Block
PLC	Programmable Logic Controller
PP	Profile Position mode (Operation mode of CiA 402 drives)
SPS	Speicher Programmierbare Steuerung
EC-STA	acontis' Slave Test Application to control and diagnosis EC-Master