BECKHOFF® TwinCAT LinMot Library

E1250-EC-UC, E1130-DP-xx, E1100-CO-xx, B1100-GP-xx

Axis Control, MC Commands & Configuration Modules

Version 1.0.3 (eng) fj, February 1st, 2012

Note
The information in this documentation reflects the stage of development at the time of press and is therefore without obligation. NTI AG reserves itself the right to make changes at any time and without notice to reflect further technical advance or product improvement.

NTI AG
LinMot®
Haerdlistrasse 15
CH-8957 Spreitenbach

Tel.: +41 (0)56 419 91 91
Fax: +41 (0)56 419 91 92
Email: office@LinMot.com
Homepage: www.LinMot.com

Content

Document version

Version	Date	Author	Library version	Description
1.0.0	08/15/2011	fj	1.0.2	Initial version
1.0.3	02/01/2012	fj	1.0.3	Document: • No changes Library: • FB LMct_AxisCtrlE12x0 & LMct_AxisCtrlx11x0: Operation Enabled output directly mapped to StatusWord bit 0

Use of the Library

The presented library for BECKHOFF TwinCAT provides function blocks to control LinMot controllers with EtherCAT, Profibus or CANopen interface.
The library is provided by NTI AG / LinMot free of charge with no warranty updates.
Also, LinMot accepts no liability for damages that may be caused by using these examples.

Controller:	E1250-EC-UC, E1230-DP-UC, E1130-DP-xx, E1100-CO-xx, B1100-GP-xx
Classification:	[ ] LinMot internally [x] Dissemination to customers allowed
Approval:	[x] Function Block Library [ ] Use in productive environments

Recommended Documentation

Reading the following user manuals is essential to understand the communication between the PLC and the LinMot controller. The manuals are included in LinMot-Talk software or can be downloaded here:
http://www.linmot.com/index.php?id=204

E1250-EC-xx:
- LinMot-Talk 4 User Manual
- User Manual Motion Control Software E1200
- EtherCAT: User manual for E1250-EC-xx
- User manual configuration over fieldbus interface E1200 Series

E1130-DP-xx:
- LinMot-Talk 4 User Manual
- User Manual Motion Control Software E1100/B1100
- PROFIBUS: User manual for E1130-DP (-HC, -XC)
- User manual configuration over fieldbus interface E1100/B1100 Series

E1100-CO-xx und B1100-GP-xx:
- LinMot-Talk 4 User Manual
- User Manual Motion Control Software E1100/B1100
- CANopen: User Manual for E1100-CO (-HC, -XC), E1100-GP (-HC, -XC) and B1100-GP (-HC, -XC)
- User manual configuration over fieldbus interface E1100/B1100 Series

General

The LinMot controllers can be connected over different interfaces to a Beckhoff TwinCAT PLC. This library is presented to simplify the integration of the controller into the PLC program and to show general control methods.

The package includes the following function blocks and data types:

Axis Control:
- LMct_RdAxisCom
- LMct_WrAxisCom
- LMct_AxisCtrlx11x0 — E1130-DP-xx, E1100-CO-xx, B1100-GP-xx
- LMct_AxisCtrlE12x0 — E1250-EC-UC, E1230-DP-UC

MC Function Blocks (All controllers):
- LMmt_MoveAbs
- LMmt_MoveRel
- LMmt_StartCTCommand
- LMmt_Stop
- LMmt_WriteLivePar
- LMmt_GenericMC

MC Function Blocks (E1250-EC-UC, E1230-DP-UC, E1130-DP-xx, E1100-CO-xx):
- LMav_Mod16BitCTPar
- LMav_Mod32BitCTPar
- LMav_PStream
- LMav_PVStream
- LMav_RunCurve

MC Function Blocks (E1250-EC-UC, E1230-DP-UC only):
- LMav_MoveBestehorn
- LMav_MoveSin

Config Function Blocks (E1250-EC-UC, E1230-DP-UC & E1130-DP-xx only):
- LMcf_CTAccess
- LMcf_CurveAccess
- LMcf_GetErrorTxt
- LMcf_GetModUPIDList
- LMcf_ParaAccess
- LMcf_StopStartDefault
- LMcf_WriteUPIDList

Config Function Blocks CANopen (E12x0, E11x0, B1100-GP-xx)
Not part of the LinMot.lib! Import SDOACCESS_X1X00.EXP to your project.
- LMcf_SDORead
- *LMcf_SDOWrite*

Data Types:
- tstLM_AxisRef
- tenLM_AxisState
- tstLM_AxisComIn
- tstLM_AxisComOut
- tstLM_CfgCTEntry
- tstLM_CfgUPIDListEntry

The function blocks are multi instance capable.

Used software:	BECKHOFF TwinCAT	Version v2.11.0 (Build 1555)
	PLC Control	Version v2.11.0 (Build 1016)
	System Manager	Version v2.11.0 (Build 1569)

1. Hardware Configuration

1.1 TwinCAT System Manager (E1250-EC-UC)

1.1.1 Insert LinMot Controller as slave

Insert a new device by right clicking on the EtherCAT-Master (Figure 1).
The required XML-Datei can be found in the following folder (default):

C:\Program Files\LinMot\LinMot-Talk X.X Build XXXXXXXX\Firmware\Interfaces\EtherCAT\XML

This file has to be copied to the IO folder of TwinCAT:

C:\TwinCAT\Io\EtherCAT

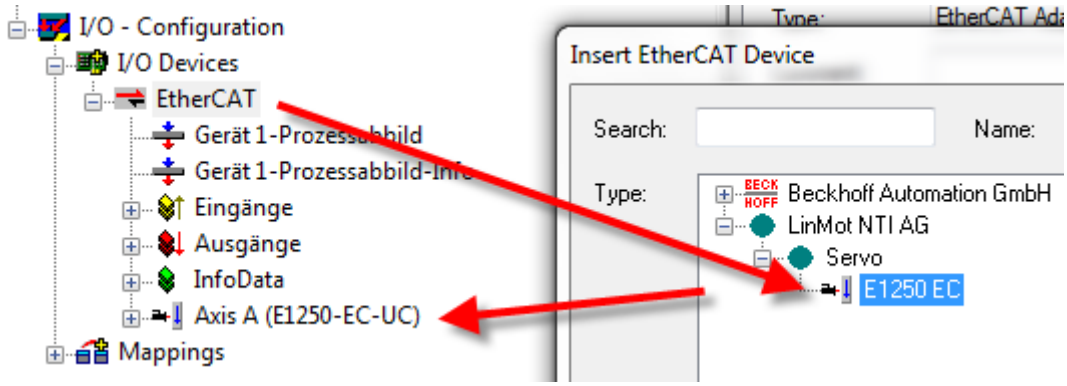


Figure 1: Insert LinMot EtherCAT slave

1.1.2 Links to PLC Control

The inputs and outputs of the modules have to be linked as shown in Figure 2. The linked variables are defined in the global variables section „Axis A Bus Communication“ of the PLC program.

Name	Linked to
StatusWord	StatusWord . Axis_A_ComIn . Inputs . Standard .
WarnWord	
DemandPosition	
ActualPosition	ComActualPosition . Axis_A_ComIn . Inputs . Sta
DemandCurrent	ComActualCurrent32 . Axis_A_ComIn . Inputs . S
Config Status Word	Status . Axis_A_ComIn . Inputs . Standard . LinMc
Config Index In	Index_In . Axis_A_ComIn . Inputs . Standard . Linl
Config Value In	Value_In . Axis_A_ComIn . Inputs . Standard . Linl
Control Word	ControlWord . Axis_A_ComOut . Outputs . Stand
Motion Command Header	MCHeader . Axis_A_ComOut . Outputs . Standar
Motion Command Par 1	MCParaDWord_00_03 . Axis_A_ComOut . Output
Motion Command Par 2	MCParaDWord_04_07 . Axis_A_ComOut . Output
Motion Command Par 3	MCParaDWord_08_11 . Axis_A_ComOut . Output
Motion Command Par 4	MCParaDWord_12_15 . Axis_A_ComOut . Output
Motion Command Par 5	MCParaDWord_16_19 . Axis_A_ComOut . Output
Config Control Word	Control . Axis_A_ComOut . Outputs . Standard . I
Config Index Out	Index_Out . Axis_A_ComOut . Outputs . Standarc
Config Value Out	Value_Out . Axis_A_ComOut . Outputs . Standarc

Figure 2: Input and output links

1.1.3 Config Module

If config function blocks are used the „Config Module“ has to be insert and linked as well.

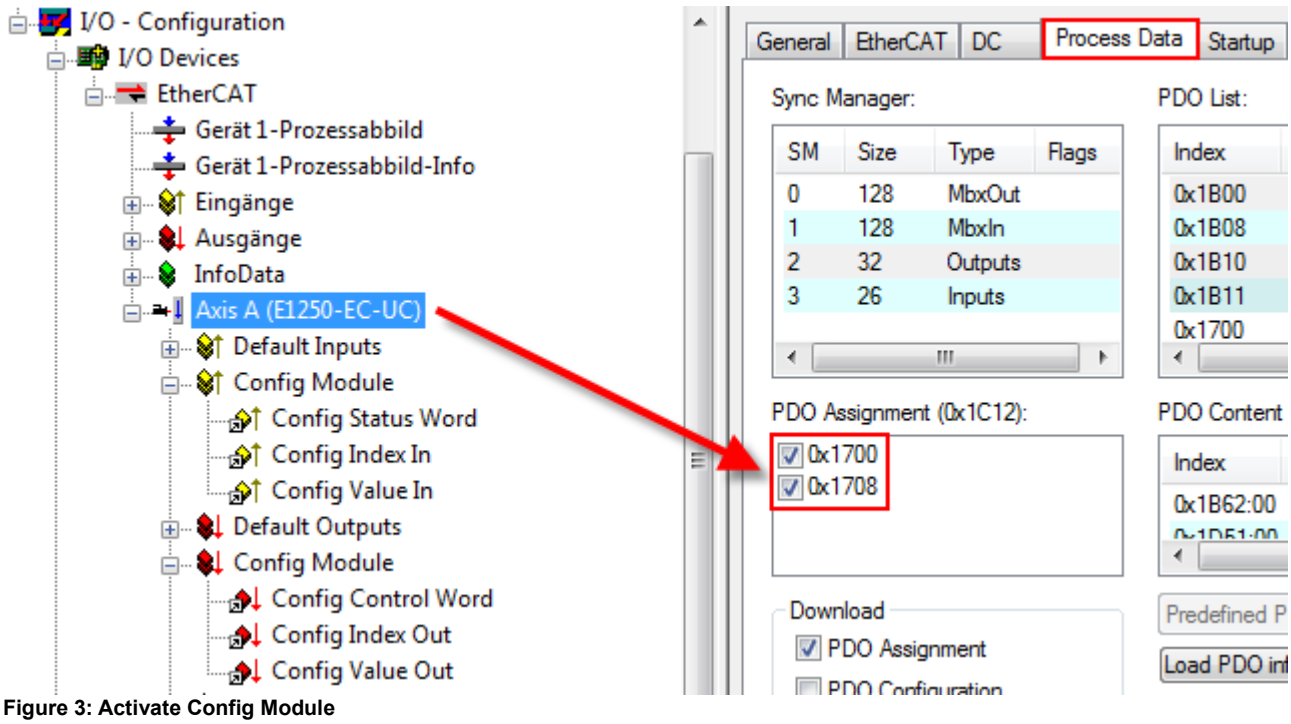


Figure 3: Activate Config Module

1.2 TwinCAT System Manager (E1130-DP-xx)

1.2.1 Insert LinMot Controller as Slave

Insert a new “Generic Profibus Box” by right clicking on the Profibus-Master (Here FC31XX, Figure 4).
The required GSD file can be found by default in the following folder:

C:\Program Files\LinMot\LinMot-Talk X.X Build XXXXXXXX\Firmware\Interfaces\Profibus\GSD\

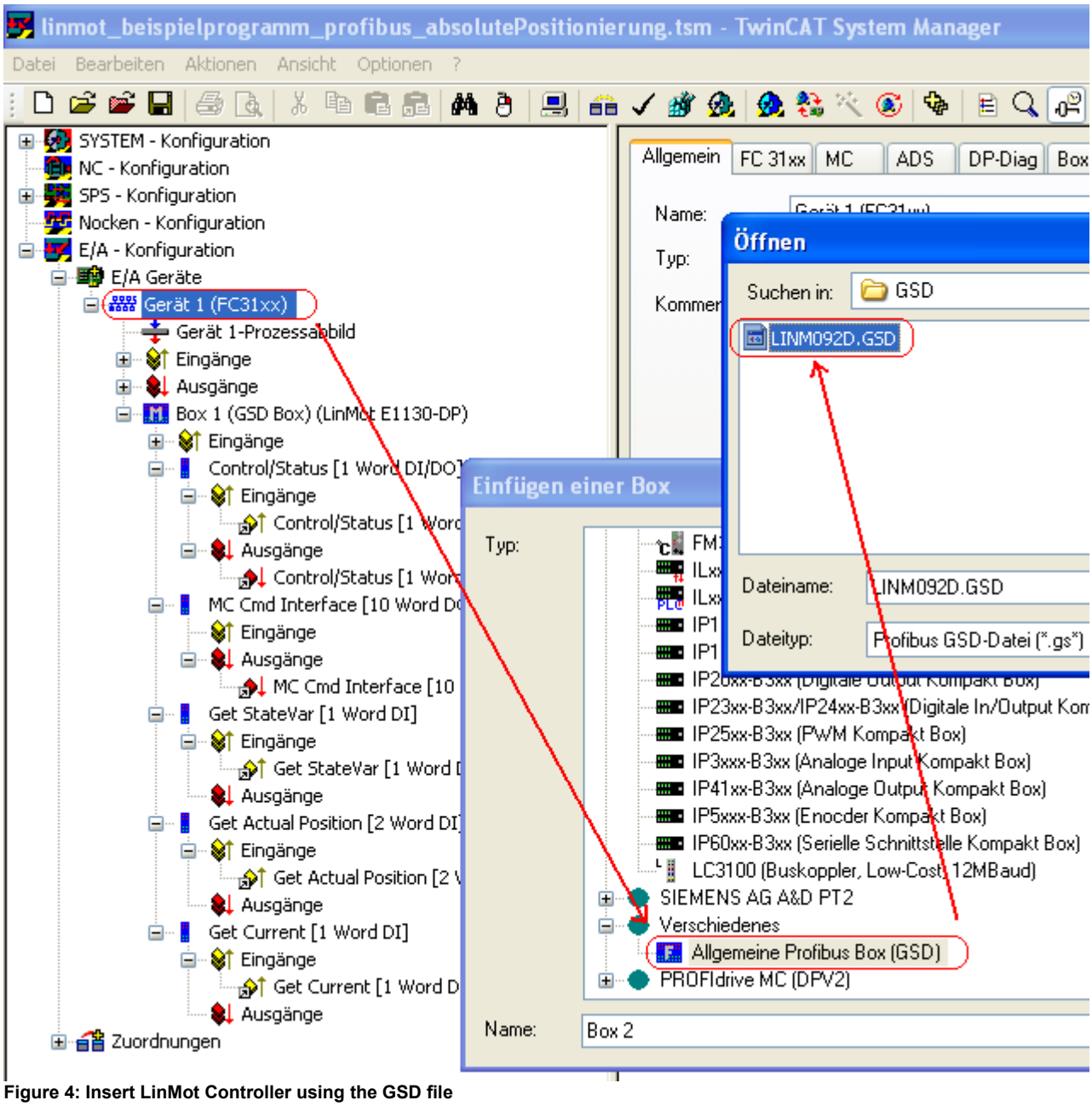


Figure 4: Insert LinMot Controller using the GSD file

Afterwards add the needed modules (Control/Status, MC Cmd Interface, Get StateVar, Get Actual Position and Get Current). Right click on Box 1 -> Append Module.

1.2.2 Links to PLC Control

The inputs and outputs of the modules have to be linked as shown in Figure 5. The linked variables are defined in the global variables section „Axis A Bus Communication“ of the PLC program.

Name	Linked to
Control/Status [1 Word DI/DO]_0_0	StatusWord . Axis_A_ComIn . Inputs . Standard . LinMot_Libra...
Get StateVar [1 Word DI]_0_0	StateVar . Axis_A_ComIn . Inputs . Standard . LinMot_Library_I...
Get Actual Position [2 Word DI]_0_0	ComActualPosition . Axis_A_ComIn . Inputs . Standard . LinM...
Get Current [1 Word DI]_0_0	ComActualCurrent16 . Axis_A_ComIn . Inputs . Standard . Lin...
Parameter Channel [4 Word DI/DO]_0_0	Status, Index_In, Value_In
Control/Status [1 Word DI/DO]_1_0	ControlWord . Axis_A_ComOut . Outputs . Standard . LinMot_...
MC Cmd Interface [10 Word DO]_1_0	MCHeader, MCParaWord0, MCParaWord1, MCParaWord2, M...
Parameter Channel [4 Word DI/DO]_1_0	Control, Index_Out, Value_Out

Figure 5: Input and output links

1.2.3 Config Module

If config function blocks are used the „Parameter Channel“ module has to be insert and linked as well.

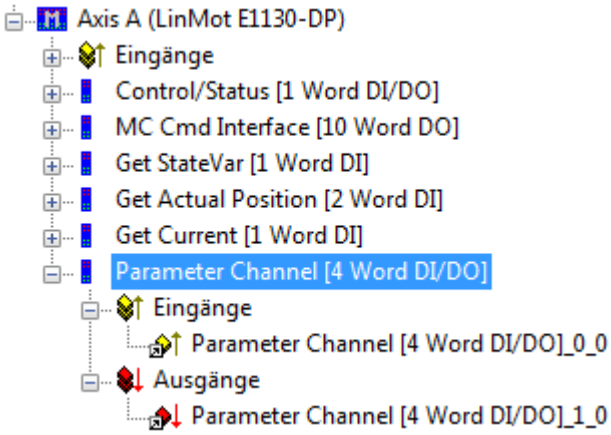


Figure 6: Module "Parameter Channel"

1.3 TwinCAT System Manager (E1100-CO-xx, B1100-GP-xx)

1.3.1 Insert LinMot Controller as slave

By right clicking on the CANopen master (Here EL6751, Figure 7) and choosing „Import Box...“ the included „Axis A (LinMot X11X0).tce“ is inserted.

The imported device already has all required links to the PLC program.
The MACID is 2 by default.

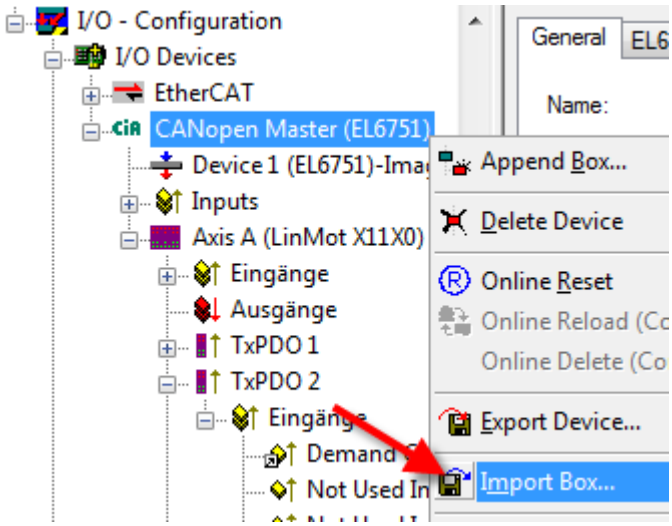


Figure 7: Import Box

A suitable EDS file is included in this library (X11X0_9W_MCI.EDS). The transmission type has to be set for each PDO manually to 1 (cyc, sync) (See Figure 8).

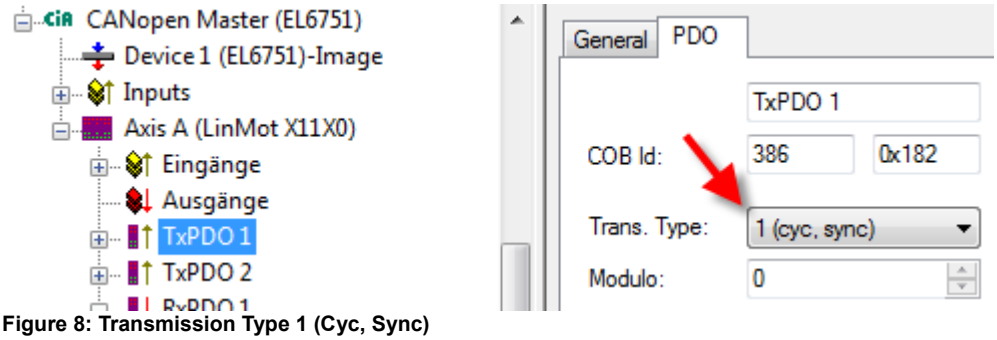


Figure 8: Transmission Type 1 (Cyc, Sync)

Note

If a box is imported as shown in Figure 7 the EDS file is not required!

1.3.2 Verknüpfungen zu PLC Control

The inputs and outputs of the PDO's have to be linked as shown in Figure 9. The linked variables are defined in the global variables section “Axis A Bus Communication” of the PLC program.

Name	Linked to
Status Word [1 W]	StatusWord . Axis_A_ComIn . Inputs . Standard . LinMot
State Var [1 W]	StateVar . Axis_A_ComIn . Inputs . Standard . LinMot_Lil
Actual Position [2 W]	ComActualPosition . Axis_A_ComIn . Inputs . Standard
Demand Current [1 W]	ComActualCurrent16 . Axis_A_ComIn . Inputs . Standar
Control Word [1 W]	ControlWord . Axis_A_ComOut . Outputs . Standard . Li
MCHeader [1 W]	MCHeader . Axis_A_ComOut . Outputs . Standard . LinM
MCParameterWord0 [1 W]	MCParaWord0 . Axis_A_ComOut . Outputs . Standard .
MCParameterWord1 [1 W]	MCParaWord1 . Axis_A_ComOut . Outputs . Standard .
MCParameterWord2 [1 W]	MCParaWord2 . Axis_A_ComOut . Outputs . Standard .
MCParameterWord3 [1 W]	MCParaWord3 . Axis_A_ComOut . Outputs . Standard .
MCParameterWord4 [1 W]	MCParaWord4 . Axis_A_ComOut . Outputs . Standard .
MCParameterWord5 [1 W]	MCParaWord5 . Axis_A_ComOut . Outputs . Standard .
MCParameterWord6 [1 W]	MCParaWord6 . Axis_A_ComOut . Outputs . Standard .
MCParameterWord7 [1 W]	MCParaWord7 . Axis_A_ComOut . Outputs . Standard .
MCParameterWord8 [1 W]	MCParaWord8 . Axis_A_ComOut . Outputs . Standard .

Figure 9: Input and output links to the PLC program

1.3.3 PDO Mapping

Figure 10 shows the mapping of the PDO's

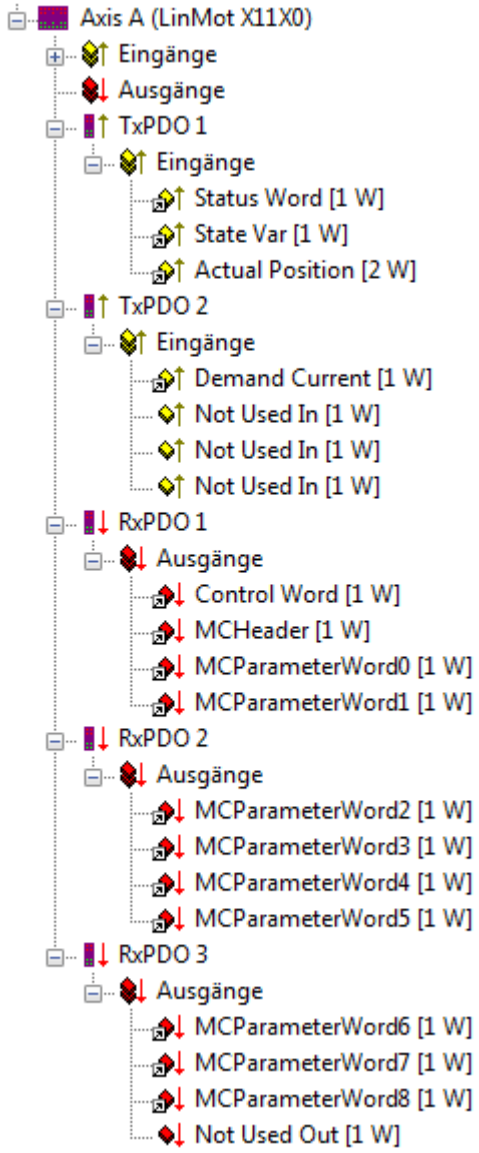


Figure 10: PDO Mapping

1.4 Configuration LinMot Controller

The LinMot Controller is configured with LinMot-Talk: http://www.linmot.com/index.php?id=204

It is assumed that the motor attached to the controller is already configured with the motor wizard.

Note

In case of doubt reset the controller to default values and afterwards configure the motor with the motor wizard. Be sure to save your actual configuration before this step!

Set controller to default values (E1xx0 series controllers only):

1. Remove 24V supply from controller.
2. Set both rotary hex switches (S1 and S2) to F.
3. Restore 24V power to controller. The ERROR and WARN LED's should blink alternately.
4. Set both rotary hex switches (S1 and S2) to 0.
5. Wait until EN and WARN led blink together.
6. Remove and restore 24V power to controller.

Alternatively for all controller types an empty configuration file can be generated with LinMot-Talk („File -> Create offline“, select required interface and/or application!). This configuration can be saved („File -> Export“) and afterwards imported to the controller.

1.4.1 E1250-EC-UC (EtherCAT)

No settings required on controller side.

1.4.2 E1130-DP-xx & E1230-DP-UC (Profibus)

The only setting that must be done on the controller according to the Profibus interface is the node address.
By default it is set with the rotary hex switches S1 (ID High) and S2 (ID Low) on the front of the controller.
Alternatively it can be set with the parameter “Node Address Parameter Value” (UPID 2076h).
This requires the parameter “Node Address Selection” (UPID 206Ch) to be set to “On”.
All other Profibus interface parameters are left to their default values.

1.4.3 E1100-CO-xx & B1100-GP-xx (CANopen)

E11x0 series controller:

- Import the CANopen interface configuration (included in the library)
  (File -> Import, E11x0_CANopen_Interface_Config.lmc)
- Set MACID using the rotary hex switch S2 on the controller

B1100-GP controller:

- Import the CANopen interface configuration (included in the library)
  (File -> Import, B1100-GP_CANopen_Interface_Config.lmc)
- Set MACID and Baud-rate
  MACID: Set UPID 6081h to the desired MACID
  Baud-rate: Set UPID 60B3h to 500 kBit/s

2. Data Types

2.1 General (Axis Control & MC Commands)

2.1.1 tstLM_Axis

The data type LM_stAxisRef contains all data required for the communication between the LinMot controller and the library function blocks.

```
TYPE tstLM_Axis :
STRUCT
(*This struct includes all required data for a proper use of the LinMot Function Blocks*)
(*General outputs to fieldbus*)
    ControlWord             :UINT;
    MCHeader                :UINT;
    MCParaWord0             :UINT;
    MCParaWord1             :UINT;
    MCParaWord2             :UINT;
    MCParaWord3             :UINT;
    MCParaWord4             :UINT;
    MCParaWord5             :UINT;
    MCParaWord6             :UINT;
    MCParaWord7             :UINT;
    MCParaWord8             :UINT;
    MCParaWord9             :UINT;
    MCParaWord10            :UINT;
    MCParaWord11            :UINT;
    MCParaWord12            :UINT;
    MCParaWord13            :UINT;
(*General inputs from fieldbus*)
    StatusWord              :UINT;
    StateVar                :UINT;
    ComActualPosition       :DINT;
    ComActualCurrent32      :DINT;  (*32Bit actual current For E1200 series controllers*)
    ComActualCurrent16      :INT;   (*16Bit actual current For x11x0 series controller*)
(*Config channel outputs to fieldbus*)
    Control                 :UINT;  (*Config Channel Control*)
    Index_Out               :UINT;  (*Argument (meaning depends on Cmd ID*)
    Value_Out               :UDINT; (*Argument (meaning depends on Cmd ID*)
(*Config channel inputs from fieldbus*)
    Status                  :UINT;  (*Config Channel Status*)
    Index_In                :UINT;  (*Argument (meaning depends on Cmd ID*)
    Value_In                :UDINT; (*Argument (meaning depends on Cmd ID*)
(*Status*)
    AxisState               :tenLM_AxisState;
    CommandRunning          :BOOL;
    CommandAborted          :BOOL;
    ConfigChannelBusy       :BOOL;
(*Information (Not required for prober work of the library function blocks)*)
    AxisName                :STRING;
    AxisNr                  :UINT;
    AxisCtrlType            :STRING;(*Eg. 'E1200-EC-UC', 'E1130-DP', 'B1100-GP', ....*)
END_STRUCT
END_TYPE
```

Figure 11: tstLM_Axis

Note

Additional information regarding the meaning of the data can be found in the user manual “Motion Control Software”. (Recommended Documentation)

2.1.2 tstLM_AxisComIn

The data type tstLM_AxisComIn contains all data sent by the controller and can be mapped directly to the system manager.

```
TYPE tstLM_AxisComIn :
STRUCT
    StatusWord : UINT; (*Statusword of the controller*)
    StateVar : UINT; (*StateVar of the controller*)
    ComActualPosition : DINT; (*32Bit actual position*)
    ComActualCurrent32 : DINT; (*32Bit actual current For E1200 series controllers*)
    ComActualCurrent16 : INT; (*16Bit actual current For x11x0 series controller*)

    Status : UINT; (*Config Channel Status*)
    Index_In : UINT; (*Argument (meaning depends on Cmd ID*)
    Value_In : UDINT; (*Argument (meaning depends on Cmd ID*)
END_STRUCT
END_TYPE
```

Figure 12: tstLM_AxisComIn

2.1.3 tstLM_AxisComOut

The data type tstLM_AxisComOut contains all data sent to the controller and can be mapped directly to the system manager.

```
TYPE tstLM_AxisComOut :
STRUCT
    ControlWord : UINT; (*ControlWord of the controller*)
    MCHeader : UINT; (*Motion command header*)
    MCParaWord0 : UINT; (*Motion command parameter word 0*)
    MCParaWord1 : UINT; (*Motion command parameter word 1*)
    MCParaWord2 : UINT; (*Motion command parameter word 2*)
    MCParaWord3 : UINT; (*Motion command parameter word 3*)
    MCParaWord4 : UINT; (*Motion command parameter word 4*)
    MCParaWord5 : UINT; (*Motion command parameter word 5*)
    MCParaWord6 : UINT; (*Motion command parameter word 6*)
    MCParaWord7 : UINT; (*Motion command parameter word 7*)
    MCParaWord8 : UINT; (*Motion command parameter word 8*)
    MCParaWord9 : UINT; (*Motion command parameter word 9*)
    MCParaWord10 : UINT; (*Motion command parameter word 10*)
    MCParaWord11 : UINT; (*Motion command parameter word 11*)
    MCParaWord12 : UINT; (*Motion command parameter word 12*)
    MCParaWord13 : UINT; (*Motion command parameter word 13*)
    MCParaDWord_00_03 : UDINT; (*Motion command parameter bytes 0-3 (used for E12X0 s
    MCParaDWord_04_07 : UDINT; (*Motion command parameter bytes 4-7 (used for E12X0 s
    MCParaDWord_08_11 : UDINT; (*Motion command parameter bytes 8-11 (used for E12X0
    MCParaDWord_12_15 : UDINT; (*Motion command parameter bytes 12-15 (used for E12X(
    MCParaDWord_16_19 : UDINT; (*Motion command parameter bytes 16-19 (used for E12X(
    MCParaDWord_20_23 : UDINT; (*Motion command parameter bytes 20-23 (used for E12X(
    MCParaDWord_24_27 : UDINT; (*Motion command parameter bytes 24-27 (used for E12X(

    Control : UINT; (*Config Channel Control*)
    Index_Out : UINT; (*Argument (meaning depends on Cmd ID*)
    Value_Out : UDINT; (*Argument (meaning depends on Cmd ID*)
END_STRUCT
END_TYPE
```

Figure 13: tstLM_AxisComOut

2.1.4 tenLM_AxisState

PLCopen like state display of the axis. No functional meaning at the moment.

```
TYPE tenLM_AxisState :
(*PLC Open like naming of the controllers state - Not proper implemented yet*)
(
    Disabled_LM,
    StandStill_LM,
    Homing_LM,
    ErrorStop_LM,
    Stopping_LM,
    DiscreteMotion_LM,
    ContinuousMotion_LM,
    SynchronizedMotion_LM,
    Jogging_LM
);
END_TYPE
```

Figure 14: tenLM_AxisState

2.2 Data Types of the Config Function Blocks

2.2.1 tstLM_CfgCTEntry

The data type tstLM_CfgCTEntry contains the data of one command table entry.
Is used by:

- LMcf_CTAccess

E1250-EC-UC, E1230-DP-UC & E1130-DP-xx Controllers only!

```
TYPE LM_stConfigCTEntry :
STRUCT
    CTID        :UINT;                      (*ID of command table entry*)
    BlockSize   :UINT;                      (*Size of EntryBlock in Bytes*)
    DataBlock   :ARRAY[0..15] OF UDINT;     (*EntryBlock, 64Bytes*)

(* Example of a Command Table Entry Data Block:
0002A701h,  LowWord = Entry Version ID (fix A701h) / HighWord = Sequenced Entry ID
00000100h,  LowWord: Motion Cmd Header(010xh=VAI Go To Pos) / HighWord: Motion Cmd Par Byte 0..1
42400000h,  Motion Cmd Par Byte 2..5
4240000Fh,  Motion Cmd Par Byte 6..9
4240000Fh,  Motion Cmd Par Byte 10..13
0000000Fh,  Motion Cmd Par Byte 14..17
00000000h,  Motion Cmd Par Byte 18..21
00000000h,  Motion Cmd Par Byte 22..25
00000000h,  Motion Cmd Par Byte 26..29
6E550000h,  LowWord: Motion Cmd Par Byte 30..31 / HighWord: Entry Name Character 0..1
656D616Eh,  Entry Name Character 2..5
00000064h,  Entry Name Character 6..9
00000000h,  Entry Name Character 10..13
FFFF0000h,  LowWord: Entry Name Character 14..15 / HighWord: Reserved for future use
FFFFFFFFh,  Reserved for future use
FFFFFFFFh,  Reserved for future use
*)
END_STRUCT
END_TYPE
```

Figure 15: tstLM_CfgCTEntry

Note

Additional information can be found in the user manual “LinMot Controller Configuration over Fieldbus Interfaces” (Recommended Documentation).

2.2.2 tstLM_CfgUPIDListEntry

tstLM_CfgUPIDListEntry contains the number and value of a LinMot parameter.
Used in:

- LMcf_GetModUPIDList
- LMcf_WriteUPIDList

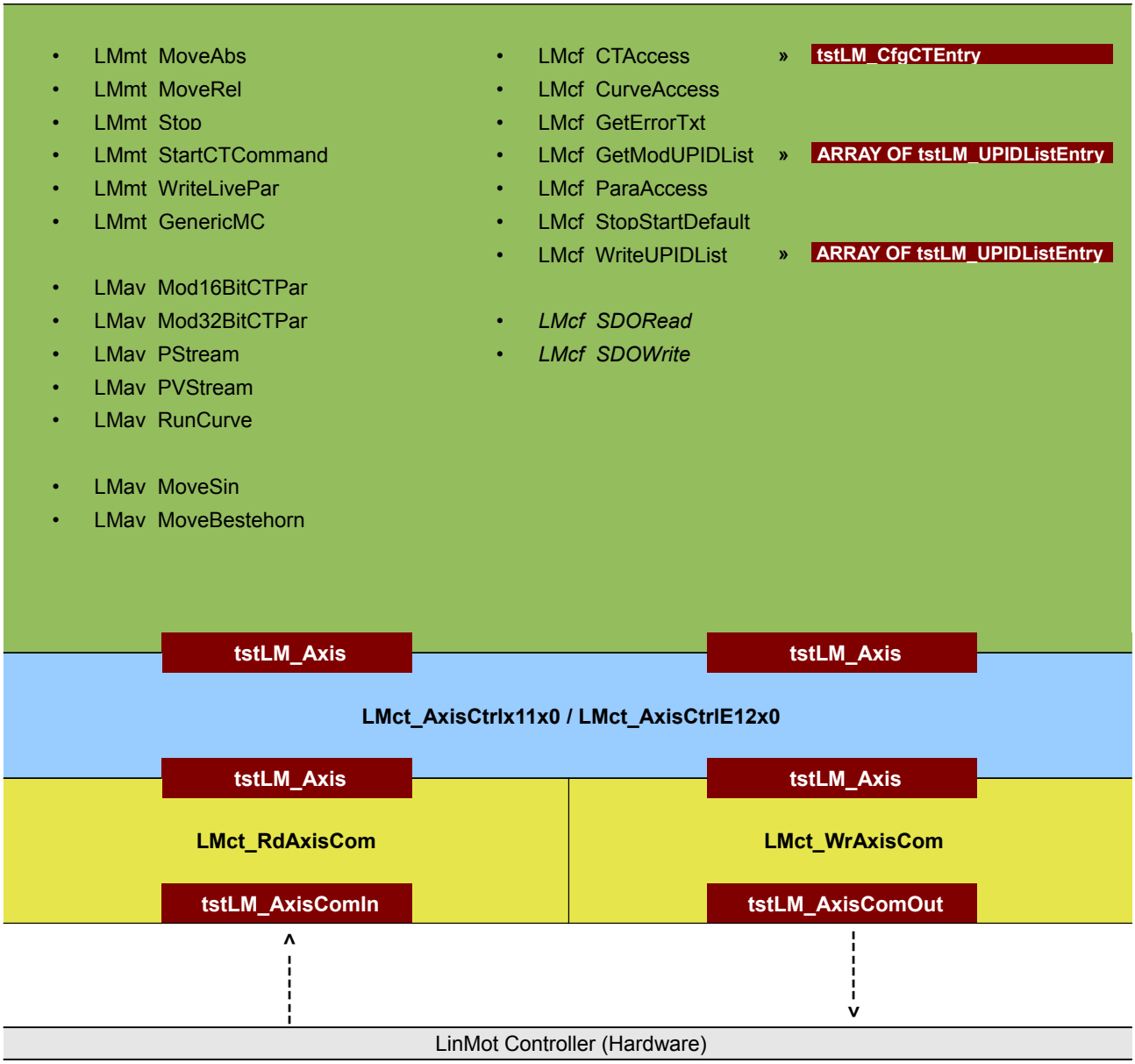
E1250-EC-UC, E1230-DP-UC & E1130-DP-xx Controllers only!

```
TYPE LM_stConfigUPIDList :
STRUCT
    (*Array size may have to be increased depending of the number of UPIDs used*)
    UPIDListEntries          :ARRAY[0..2000] OF LM_stConfigUPIDListEntry;
END_STRUCT
END_TYPE
```

Figure 16: tstLM_CfgUPIDListEntry

3. Function Blocks

3.1 Overview and dependencies



3.2 IO and Axis Control

3.2.1 LMct_RdAxisCom

This function block LMct_RdAxisCom reads the input data and puts them to the axis data type.
Should be called at the beginning of the PLC cycle or at least before all other library function blocks

Supported controllers:
- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen)*
- *B1100-GP-xx (CANopen)*

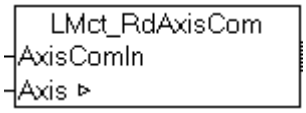


Figure 17: LMct_RdAxisCom

Inputs		
Name	**Data type**	**Description**
AxisComIn	tstLM_AxisComIn	Fieldbus input data
Axis	tstLM_Axis	Axis reference (IN_OUT)

3.2.2 LMct_WrAxisCom

This function block LMct_WrAxisCom reads the output data of the axis reference (tstLM_Axis), prepares and writes them to the output data struct (tstLM_AxisComOut).
Should be called at the end of the PLC cycle or at least after all other library function blocks.

Supported controllers:
- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen)*
- *B1100-GP-xx (CANopen)*

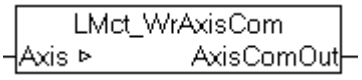


Abbildung 18: LMct_WrAxisCom

Inputs		
Name	**Data type**	**Description**
Axis	tstLM_Axis	Axis reference (IN_OUT)

Outputs		
Name	**Data type**	**Description**
AxisComOut	tstLM_AxisComOut	Fieldbus output data

3.2.3 LMct_AxisCtrlE12x0

This function block is used to control the state machine of a LinMot controller. The outputs show the status of the controller.

Supported controllers:
- *E1250-EC-UC & E1230-DP-UC*

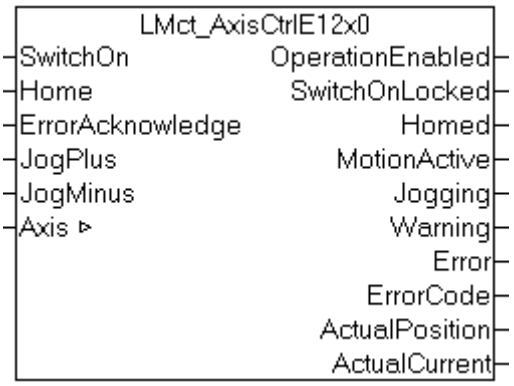


Figure 19: LMct_AxisCtrlE12x0

3.2.4 LMct_AxisCtrlx11x0

This function block is used to control the state machine of a LinMot controller. The outputs show the status of the controller.

Supported controllers:
- *E1130-DP-xx, E1100-CO-xx, B1100-GP-xx*

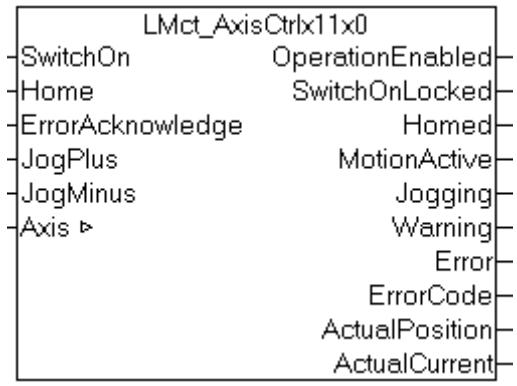


Figure 20: LMct_AxisCtrlx11x0

3.2.5 Inputs and Outputs of the Axis Control function blocks

Both function blocks have the same inputs and outputs.
The functionality is identical.

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
SwitchOn	Bool		Switch on axis
Home	Bool		Start homing of the axis (Has to stay TRUE until the output Homed is set)
ErrorAcknowledge	Bool		Error acknowledge on rising edge
JogPlus	Bool		Jog move positive
JogMinus	Bool		Jog move negative

Table 1: Inputs LMct_AxisCtrlE12x0 & LMct_AxisCtrlx11x0

Outputs		
Name	**Data type**	**Description**
OperationEnabled	Bool	Axis is powered and ready for commands
SwitchOnLocked	Bool	Switch on is locked (-> Release SwitchOn)
Homed	Bool	Axis is homed (has reference)
MotionActive	Bool	Setpoint generation (VAI, curve) active (the controller is attempting to move)
Jogging	Bool	Axis is moving in jog mode
Warning	Bool	Warning active
Error	Bool	Error has occurred and controller is in the error state
ErrorCode	UINT	Shows the error code. (See user manual „Motion Control SW“)
ActualPosition	Real	Actual position of the axis in mm
ActualCurrent	Real	Actual current of the axis in A (Ampère)

Table 2: Outputs LMct_AxisCtrlE12x0 & LMct_AxisCtrlx11x0

Note

Additional information including the state machine of a LinMot controller can be found in the user manual “Motion Control SW” (Recommended Documentation).

Attention

All IO and axis control function blocks must be called cyclically!

3.3 MC Function Blocks (E12x0, E11x0, B1100-GP Controller)

3.3.1 LMmt_MoveAbs

With this function block a motion to an absolute position with the set maximal velocity, acceleration and deceleration is executed.

Supported controllers:

- *E12x0 series (EtherCAT oor Profibus)*
- *E11x0 series (Profibus or CANopen)*
- *B1100-GP-xx (CANopen)*

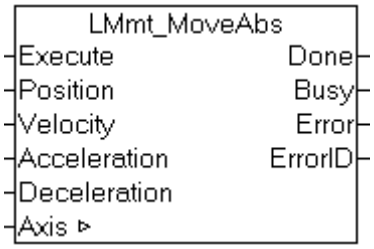


Figure 21: LMmt_MoveAbs

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
Position	Real		Target position in [mm]
Velocity	Real		Max. velocity in [m/s]
Acceleration	Real		Acceleration in [m/s^2]
Deceleration	Real		Deceleration in [m/s^2]

Table 3: Inputs LMmt_MoveAbs

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command done and axis in target position
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 4: Outputs LMmt_MoveAbs

3.3.2 LMmt_MoveRel

With this function block a relative motion can be done. The motions dynamics are defined with the inputs velocity, acceleration and deceleration.

Supported controllers:

- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen)*
- *B1100-GP-xx (CANopen)*

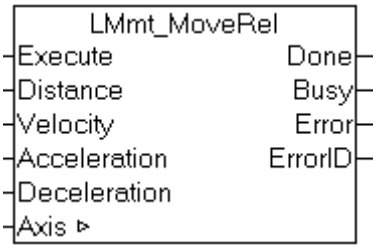


Figure 22: LMmt_MoveRel

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
Distance	Real		Position increment in [mm]
Velocity	Real		Max. velocity in [m/s]
Acceleration	Real		Acceleration in [m/s^2]
Deceleration	Real		Deceleration in [m/s^2]

Table 5: Inputs LMmt_MoveRel

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command done and axis in target position
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 6: Outputs LMmt_MoveRel

3.3.3 LMmt_StartCTCommand

This function block starts a line of the Command Table (stored in the controller).

Supported controllers:

- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen)*
- *B1100-GP-xx (CANopen)*

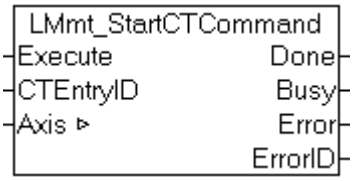


Figure 23: LMmt_StartCTCommand

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
CTEntryID	UINT	1...255 (31)	ID of the line of the Command Table (Attention: on B1100-GP-xx max. 31)

Table 7: Inputs LMmt_StartCTCommand

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command sent
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 8: Outputs LMmt_StartCTCommand

3.3.4 LMmt_Stop

This function block stops the axis immediately with the set deceleration.
Other active MC function blocks are aborted!

Supported controllers:

- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen)*
- *B1100-GP-xx (CANopen)*

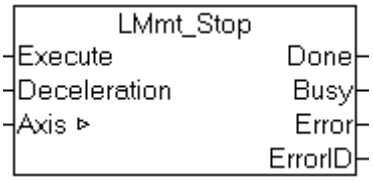


Figure 24: LMmt_Stop

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
Deceleration	Real		Deceleration in [m/s²]

Table 9: Inputs LMmt_Stop

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command sent and axis stopped
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 10: Outputs LMmt_Stop

3.3.5 LMmt_WriteLivePar

With this function block a Live Parameter of the controller can be modified/written (“live” parameters can be changed during runtime).

Supported controllers:

- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen)*
- *B1100-GP-xx (CANopen)*

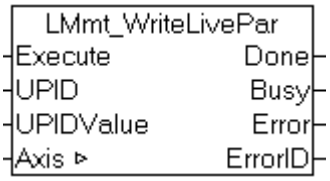


Figure 25: LMmt_WriteLivePar

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
UPID	UINT		Parameter address (Unique Parameter ID)
UPIDValue	DINT		Parameter value

Table 11: Inputs LMmt_WriteLivePar

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command sent
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 12: Outputs LMmt_WriteLivePar

3.3.6 LMmt_GenericMC

With this function block all available Motion Control commands (of the used controller) can be executed.
The parameters have to be scaled according to the selected MCHeader!
A list of all supported motion commands can be found in the user manual “Motion Control SW”.

Supported controllers:

- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen)*
- *B1100-GP-xx (CANopen)*

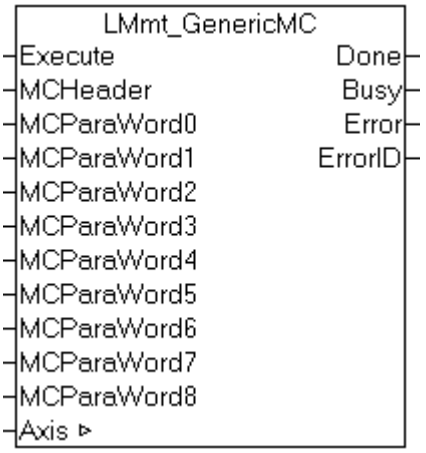


Figure 26: LMmt_GenericMC

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
MCHeader	UINT		Motion command header
MCParaWord0	UINT		0. Parameter word
MCParaWord1	UINT		1. Parameter word
MCParaWord2	UINT		2. Parameter word
MCParaWord3	UINT		3. Parameter word
MCParaWord4	UINT		4. Parameter word
MCParaWord5	UINT		5. Parameter word
MCParaWord6	UINT		6. Parameter word
MCParaWord7	UINT		7. Parameter word
MCParaWord8	UINT		8. Parameter word

Table 13: Inputs LMmt_GenericMC

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command sent
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 14: Outputs LMmt_GenericMC

3.4 MC Function Blocks (E12x0, E11x0 Controller)

3.4.1 LMav_Mod16BitCTPar

With this function block the value of a parameter (16Bit) in the command table can be modified (RAM only).

Supported controllers:
- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen)*

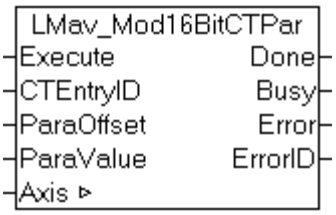


Figure 27: LMav_Mod16BitCTPar

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
CTEntryID	UINT		ID of the command table line
ParaOffset	UINT		Offset of the parameter to be written
ParaValue	INT		Value of the parameter to be written

Table 15: Inputs LMav_Mod16BitCTPar

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command sent
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 16: Outputs LMav_Mod16BitCTPar

Note

Additional information for this command can be found in the user manual “Motion Control Software”. (Recommended Documentation)

3.4.2 LMav_Mod32BitCTPar

With this function block the value of a parameter (32Bit) in the command table can be modified (RAM only).

Supported controllers:
- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen)*

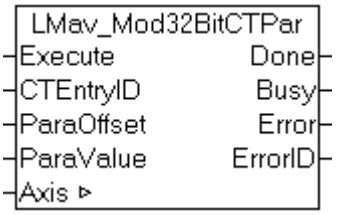


Figure 28: LMav_Mod32BitCTPar

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
CTEntryID	UINT		ID of the command table line
ParaOffset	UINT		Offset of the parameter to be written
ParaValue	DINT		Value of the parameter to be written

Table 17: Inputs LMav_Mod32BitCTPar

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command sent
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 18: Outputs LMav_Mod32BitCTPar

Note

Additional information for this command can be found in the user manual “Motion Control Software”. (Recommended Documentation)

3.4.3 LMav_RunCurve

With this function block a motion profile (curve) that is stored in the controller can be executed.

Supported controllers:

- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen)*

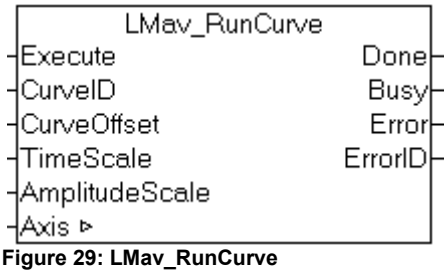


Figure 29: LMav_RunCurve

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
CurveID	UINT	1...99	Curve number (ID)
CurveOffset	Real		Offset of the curve
TimeScale	Real	0...200	Time scale in [%]
AmplitudeScale	Real	-2000...+2000	Amplitude scale in [%]

Table 19: Inputs LMav_RunCurve

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command executed and axis in target position
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 20: Outputs LMav_RunCurve

Note

Additional information for this command can be found in the user manual “Motion Control Software”. (Recommended Documentation)

3.5 MC Function Blocks (E12x0 Controller)

3.5.1 LMav_MoveBestehorn

With this function block the axis can be moved to the target position using a bestehorn profile.

Supported controllers:

- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen **with installed "Sinoide" application**)*

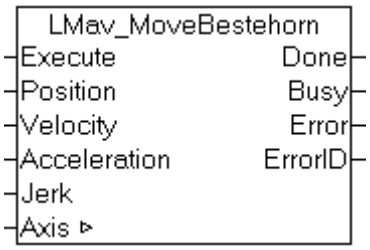


Figure 30: LMav_MoveBestehorn

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
Position	Real		Target position in [mm]
Velocity	Real		Max. velocity in [m/s]
Acceleration	Real		Acceleration in [m/s^2]
Jerk	Real		Maximum jerk in [m/s^3]

Table 21: Inputs LMav_MoveBestehorn

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command done and axis in target position
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 22: Outputs LMav_MoveBestehorn

Note

Additional information for this command can be found in the user manual "Motion Control Software". (Recommended Documentation)

3.5.2 LMav_MoveSin

With this function block the axis can be moved to the target position using a sin profile.

Supported controllers:

- *E12x0 series (EtherCAT or Profibus)*
- *E11x0 series (Profibus or CANopen **with installed "Sinoide" application**)*

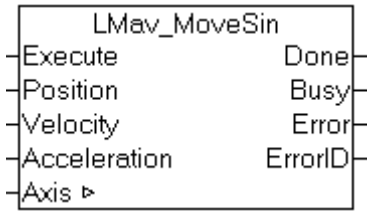


Figure 31: LMav_MoveSin

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
Position	Real		Target position in [mm]
Velocity	Real		Max. velocity in [m/s]
Acceleration	Real		Acceleration in [m/s^2]

Table 23: Inputs LMav_MoveSin

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command done and axis in target position
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 24: Outputs LMav_MoveSin

Note

Additional information for this command can be found in the user manual "Motion Control Software". (Recommended Documentation)

3.6 Config Function Blocks (E12x0, E1130-DP)

The config function blocks grant access to parameters, curves and the command table of a LinMot controller. Additionally they provide functions like restart or stop the firmware or parts of it, reading error texts (STRING) and restoring the parameters of each firmware layer to default values.

The function blocks listed in this chapter are compatible with the following controllers and interfaces:

- E1250-EC-UC EtherCAT
- E1230-DP-UC Profibus
- E1130-DP-xx Profibus

Attention

If data on the controller (command table, curves) is saved from the RAM to the flash memory the firmware layer MC_SW must be stopped! That can be done using the config function block **LMcf_StopStartDefault** with **Mode 5**. With **Mode 6** it can be restarted afterwards.

3.6.1 LMcf_StopStartDefault

This function block provides the functionality to restart the controller, stop and start single firmware layers, or set the parameters of each firmware layer to factory defaults.

Supported controllers:
- *E12x0 series (EtherCAT or Profibus)*
- *E1130-DP-xx (Profibus)*

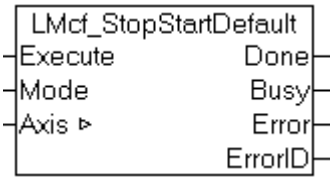


Figure 32: LMcf_StopStartDefault

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
Mode	UINT	0...6	Mode

Table 25: Inputs LMcf_StopStartDefault

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command executed
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 26: Outputs LMcf_StopStartDefault

Mode		
Value	**Used Inputs**	**Description**
0	-	Restart Controller
1	-	Set parameter ROM values to default (OS SW)
2	-	Set parameter ROM values to default (MC SW)
3	-	Set parameter ROM values to default (Interface SW)
4	-	Set parameter ROM values to default (Application SW)
5	-	Stop MC and Application Software (for Flash Access)
6	-	Start MC and Application Software

Table 27: Modes of LMcf_StopStartDefault

Attention

Mode 5&6 are important when using the config function blocks LMcf_CurveAccess and LMcf_CTAccess. Before curves or command table entries are saved from RAM to the flash memory of the controller the **MC_SW must be stopped!**

3.6.2 LMcf_CTAccess

This function block provides access to the command table of a LinMot controller. Read and write entries, delete entries, delete the complete command table, save the command table from RAM to flash memory.

Supported controllers:
- *E12x0 series (EtherCAT or Profibus)*
- *E1130-DP-xx (Profibus)*

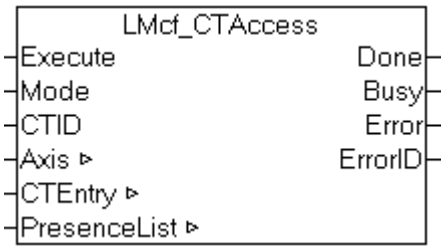


Figure 33: LMcf_CTAccess

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
Mode	UINT	0...5	Mode
CTID	UINT	1...255	Command Table ID (line number)
CTEntry	tstLM_CfgCTEntry		Command Table entry (IN_OUT)
PresenceList	Array[0..7] of UDINT		Presence List (IN_OUT)

Table 28: Inputs LMcf_CTAccess

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command executed
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 29: Outputs LMcf_CTAccess

Mode		
Value	**Used Inputs**	**Description**
0	-	Save to Flash: Saves the Command Table to the flash memory **MC_SW must be stopped!**
1	-	Delete all Entries (RAM)
2	CTID	Delete Entry (RAM)
3	CTEntry	Write Entry (ID is in CTEntry) (RAM)
4	CTID, CTEntry	Get Entry (is stored in CTEntry)
5	PresenceList	Read presence list from controller

Table 30: Modes of LMcf_CTAccess

3.6.3 LMcf_CurveAccess

This function provides access to motion profiles (curves) on the a LinMot controller. It is possible to read, write, modify and delete curves as well as saving all curves from the RAM to the flash memory.

Supported controllers:
- *E12x0 series (EtherCAT or Profibus)*
- *E1130-DP-xx (Profibus)*

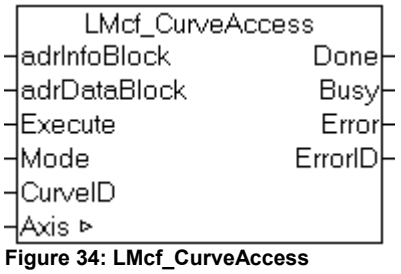


Figure 34: LMcf_CurveAccess

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
adrInfoBlock	POINTER TO ARRAY [0-18] OF UDINT		Pointer to array of UDINT which contains the curve data (start of InfoBlock)
adrDataBlock	POINTER TO ARRAY OF UDINT		Pointer to array of UDINT which contains the curve data (start of DataBlock)
Execute	Bool		Execute command (rising edge)
Mode	UINT	0...4	Mode
CurveID	UINT	1...99	Curve number (ID)

Table 31: Inputs LMcf_CurveAccess

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command executed
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 32: Outputs LMcf_CurveAccess

Mode		
Value	**Used Inputs**	**Description**
0	-	Save all Curves from RAM to Flash **MC_SW must be stopped!**
1	-	Delete all Curves (RAM)
2	adrInfoBlock, adrDataBlockt	Add Curve (RAM)
3	adrInfoBlock, adrDataBlock	Modify Curve (RAM)
4	CurveID, adrInfoBlock, adrDataBlock	Get Curve (data is stored in in the array at “adrInfoBlock”)

Table 33: Modes of LMcf_CurveAccess

Attention

The size of the ARRAY OF UDINT has to be big enough!
Example call: LMcf_CurveAccess(adrInfoBlock:= ADR(Axis_A_Curve[0]),
adrDataBlock:= ADR(Axis_A_Curve[20]),

3.6.4 LMcf_ParaAccess

This function block provides access to the parameters of a LinMot controller. Read and write RAM and ROM parameters. Read minimal, maximal and default values of a parameter.

Supported controllers:
- *E12x0 series (EtherCAT or Profibus)*
- *E1130-DP-xx (Profibus)*

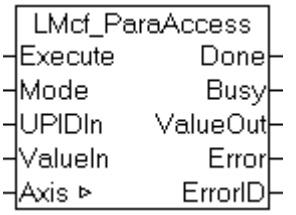


Figure 35: LMcf_ParaAccess

Inputs			
Name	**Data type**	**Range**	**Description**
Axis	tstLM_Axis		Axis reference (IN_OUT)
Execute	Bool		Execute command (rising edge)
Mode	UINT	0...7	Mode
UPIDIn	UINT		Parameter ID (UPID)
ValueIn	DINT		Value to be written

Table 34: Inputs LMcf_ParaAccess

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command executed
Busy	Bool	Command active
ValueOut	DINT	Read value / Feedback of written value
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 35: Outputs LMcf_ParaAccess

Mode		
Value	**Used Inputs**	**Description**
0	UPIDIn	Read ROM Value of Parameter by UPID
1	UPIDIn	Read RAM Value of Parameter by UPID
2	UPIDIn, ValueIn	Write ROM Value of Parameter by UPID
3	UPIDIn, ValueIn	Write RAM Value of Parameter by UPID
4	UPIDIn, ValueIn	Write RAM and ROM Value of Parameter by UPID
5	UPIDIn	Get minimal Value of Parameter by UPID
6	UPIDIn	Get maximal Value of Parameter by UPID
7	UPIDIn	Get default Value of Parameter by UPID

Table 36: Modes of LMcf_ParaAccess

3.6.5 LMcf_GetModUPIDList

This function block reads a list of parameters and their values that have been modified (compared to factory defaults). That may be used to save the configuration of a controller on the PLC.

Supported controllers:
- *E12x0 series (EtherCAT or Profibus)*
- *E1130-DP-xx (Profibus)*

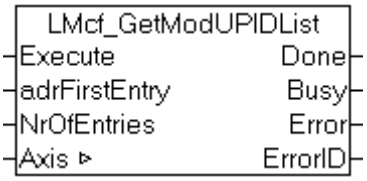


Figure 36: LMcf_GetModUPIDList

Inputs		
Name	**Data type**	**Description**
Axis	tstLM_Axis	Axis reference (IN_OUT)
Execute	Bool	Execute command (rising edge)
adrFirstEntry	POINTER TO tstLM_CfgUPIDListEntry	Pointer to the first entry of an array of tstLM_CfgUPIDListEntry
NrOfEntries	UINT	Number of entries to be read (>0)

Table 37: Inputs LMcf_GetModUPIDList

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command executed / UPID list written
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 38: Outputs LMcf_GetModUPIDList

Attention

The size of the array of tstLM_CfgUPIDListEntry has to be enough big.
Example call: LMcf_GetModUPIDList(
adrFirstEntry:= ADR(Axis_A_ModUPIDList[1]),
Axis:=Axis_A_Axis
);

3.6.6 LMcf_WriteUPIDList

This function block writes a list of parameters (UPID, Value) to the controller.

Supported controllers:

- *E12x0 series (EtherCAT or Profibus)*
- *E1130-DP-xx (Profibus)*

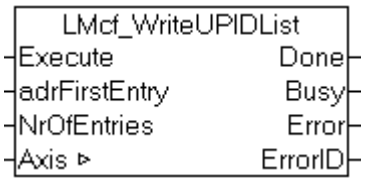


Figure 37: LMcf_WriteUPIDList

Inputs		
Name	**Data type**	**Description**
Axis	tstLM_Axis	Axis reference (IN_OUT)
Execute	Bool	Execute command (rising edge)
adrFirstEntry	POINTER TO tstLM_CfgUPIDListEntry	Pointer to the first entry of an array of tstLM_CfgUPIDListEntry
NrOfEntries	UINT	Number of entries to be written (>0)

Table 39: Inputs LMcf_WriteUPIDList

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command executed / UPID list read
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 40: Outputs LMcf_WriteUPIDList

Attention

The function block stops writing as soon as it finds a UPID number in the array that is zero **or** after the number at “NrOfEntries”.
Therefore a parameter with UPID = 0 must follow the last valid parameter in the array or the “NrOfEntries” has to be exact!

3.6.7 LMcf_GetErrorTxt

This function blocks returns a STRING containing the error text according to the input ErrorCode.

Supported controllers:

- *E12x0 series (EtherCAT or Profibus)*
- *E1130-DP-xx (Profibus)*

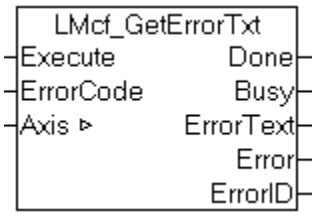


Figure 38: LMcf_GetErrorTxt

Inputs		
Name	**Data type**	**Description**
Axis	tstLM_Axis	Axis reference (IN_OUT)
Execute	Bool	Execute command (rising edge)
ErrorCode	UINT	Error code (See axis control function block)

Table 41: Inputs LMcf_GetErrorTxt

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command executed / UPID list read
Busy	Bool	Command active
ErrorText	String	Error text as STRING
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (See chapter 4. Error Descriptions)

Table 42: Outputs LMcf_GetErrorTxt

3.7 Config Function Blocks CANopen (E11x0, B1100-GP Controller)

3.7.1 LMcf_SDORead

*Import the export file **SDOACCESS_X1X00.EXP** to your project to be able to use this function block!*

This function block sends an SDO read request to the LinMot controller.

Supported controllers:
- *E11x0 series (CANopen)*
- *B1100-GP-xx (CANopen)*

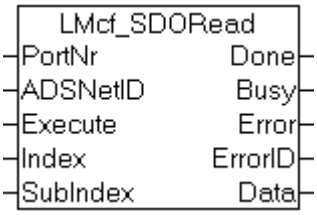


Figure 39: LMcf_SDORead

Inputs			
Name	**Data type**	**Range**	**Description**
PortNr	UINT		AMS port number (MACID of the controllers)
ADSNetID	String		ADSNetID of the CANopen masters
Execute	Bool		Execute command (rising edge)
Index	UINT		Object Dictionary Index
SubIndex	USINT		Object Dictionary SubIndex

Table 43: Inputs LMcf_SDORead

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command executed / Data read
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (see Beckhoff ADS errors)
Data	DINT	Data read

Table 44: Outputs LMcf_SDORead

Note

Additional information and the Object Dictionary can be found in the user manual *“CANopen Interface”.* (Recommended Documentation)

3.7.2 LMcf_SDOWrite

*Import the export file **SDOACCESS_X1X00.EXP** to your project to be able to use this function block!*

This function block sends an SDO write request to the LinMot controller.

Supported controllers:
- *E11x0 series (CANopen)*
- *B1100-GP-xx (CANopen)*

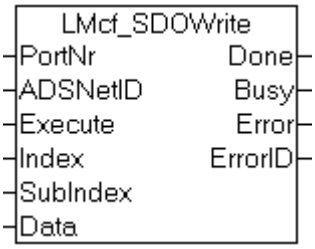


Figure 40: LMcf_SDOWrite

Inputs			
Name	**Data type**	**Range**	**Description**
PortNr	UINT		AMS port number (MACID of the controllers)
ADSNetID	String		ADSNetID of the CANopen masters
Execute	Bool		Execute command (rising edge)
Index	UINT		Object Dictionary Index
SubIndex	USINT		Object Dictionary SubIndex
Data	DINT		Data to be written

Table 45: Inputs LMcf_SDOWrite

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command executed / Data sent
Busy	Bool	Command active
Error	Bool	Error in function block
ErrorID	UINT	ErrorID (see Beckhoff ADS errors)

Table 46: OutputsLMcf_SDOWrite

Note

Additional information and the Object Dictionary can be found in the user manual *“CANopen Interface”.* (Recommended Documentation)

4. Error Descriptions

4.1 ErrorCodes of the Axis Control Function Blocks

A list of error codes can be found in the user manual „Motion Control SW“ and also in the user manual for each individual interface (Recommended Documentation).

4.2 Error ID's of the MC Function Blocks

Error ID	Error text	Description
01h	Axis not ready	Axis is not ready for motion commands. Check axis control function block if “OperationEnabled” output is set TRUE
02h	Axis already has command running	Axis has a running command. Check if another MC function block is busy. **Note:** By resetting the SwitchOn input on the axis control function block the CommandRunning flag is reset in the axis reference
03h	Axis has error	The axis has an error. Check ErrorCode on the axis control function block
04h	Command interrupted	Command has been interrupted (axis is not “OperationEnabled” any more)
05h	Command aborted	Command has been aborted (e.g. function block LinMotFB_Stop)

Table 47: Error ID's of the MC function blocks

4.3 Error ID's of the Config Function Blocks

Error ID	Error text	Description
01h	TimeOut (No response from controller)	Controller is not responding within the requested time. Check fieldbus connection
02h	ConfigChannel already busy	Config channel is already busy. Check if another config function block is busy
03h	Invalid Mode selected	Invalid mode selected. Please check Mode input
04h	Address of adrDataBlock or adrInfoBlock is invalid (0), check inputs (CurveAccess)	
05h	NrOfEntries must be greater than 0 (zero)! (UPID List)	
06h	Address of UPIDListEntry array is zero! Check adrFirstEntry input! (UPID List)	
C0h	UPID error	
C1h	Parameter Type Error	
C2h	Range Error	The value to be written is outside the parameters range
C3h	Address Usage Error	There is an attempt to write a read only parameter
C5h	Error: Command 21h	
D0h	Odd Address	
D1h	Size Error (Curve Service)	
D4h	Curve already defined / Curve not present (Curve Service)	
>D4h		Contact LinMot technical support

Table 48: Error ID's of the Config Function Blocks

Contact

SWITZERLAND

NTI AG
Haerdlistr. 15
CH-8957 Spreitenbach

Sales and Administration:	+41-(0)56-419 91 91 office@linmot.com
Tech. Support:	+41-(0)56-544 71 00 support@linmot.com
Tech. Support (Skype):	skype:support.linmot
Fax:	+41-(0)56-419 91 92
Web:	http://www.linmot.com/

USA

LinMot, Inc.
5750 Townline Road
Elkhorn, WI 53121

Sales and Administration:	877-546-3270 262-743-2555
Tech. Support:	877-804-0718 262-743-1284
Fax:	800-463-8708 262-723-6688
E-Mail:	us-sales@linmot.com
Web:	http://www.linmot-usa.com/

Please visit http://www.linmot.com/ to find the distributor near you.

Smart solutions are…

driven by LinMot® www.LinMot.com