Siemens® S7 LinMot Library

E1230-DP-UC, E1130-DP-xx

Axis Control, MC Commands & Configuration Modules

Version 1.0.2 (eng) fj, January 25th 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
1.0.1	01/24/2012	fj	1.0.1	Initial version
1.0.2	01/25/2012	fj	1.0.2	Document: • No changes Library: • Function Block LMct_AxisCtrl: Operation Enabled output directly mapped to StatusWord bit 0

Use of the Library

The presented library for Siemens® S7 provides function blocks to control LinMot controllers over Profibus interface.
The library is provided by NTI AG / LinMot free of charge with no warranty for updates.
Also, LinMot accepts no liability for damages that may be caused by using these library.

Controller:	E1230-DP-UC, E1130-DP-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 the 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
- User manual PROFIBUS E1100
- User manual configuration over fieldbus interface E1200 Series

E1130-DP-xx:
- LinMot-Talk 4 user Manual
- User manual Motion Control Software E1100/B1100
- User manual PROFIBUS E1100
- User manual configuration over fieldbus interface E1100/B1100 Series

General

The LinMot controllers can be connected over different interfaces to a Siemens S7 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_AxisCtrl

MC Function Blocks:
- LMmt_MoveAbs
- LMmt_MoveRel
- LMmt_StartCTCommand
- LMmt_Stop
- LMmt_WriteLivePar
- LMmt_GenericMC

MC Function Blocks (Advanced):
- LMav_Mod16BitCTPar
- LMav_Mod32BitCTPar
- LMav_RunCurve

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

Config Function Blocks:
- LMcf_ParaAccess
- LMcf_GetModUPIDList
- LMcf_WriteUPIDList
- LMcf_StopStartDefault
- LMcf_CurveAccess
- Lmcf_CTAccess
- LMcf_GetErrorTxt

Data Types (UDT):
- tstLM_Axis
- tstLM_CfgCTEntry
- tstLM_CfgUPIDListEntry
- tstLM_CfgCTPresenceList

The function blocks are multi instance capable.

The library was created with STEP 7 V5.4 + SP5 on a CPU 315-2 DP (Firmware V2.0)

The function blocks of this library make use of the following system functions / function blocks:
- SFB4 TON
- SFC14 DPRD_DAT
- SFC15 DPWR_DAT
- SFC20 BLKMOV

1. Hardware Configuration

1.1 Siemens HW Config

First of all install the GSD file for the controller Figure 1.
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\

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

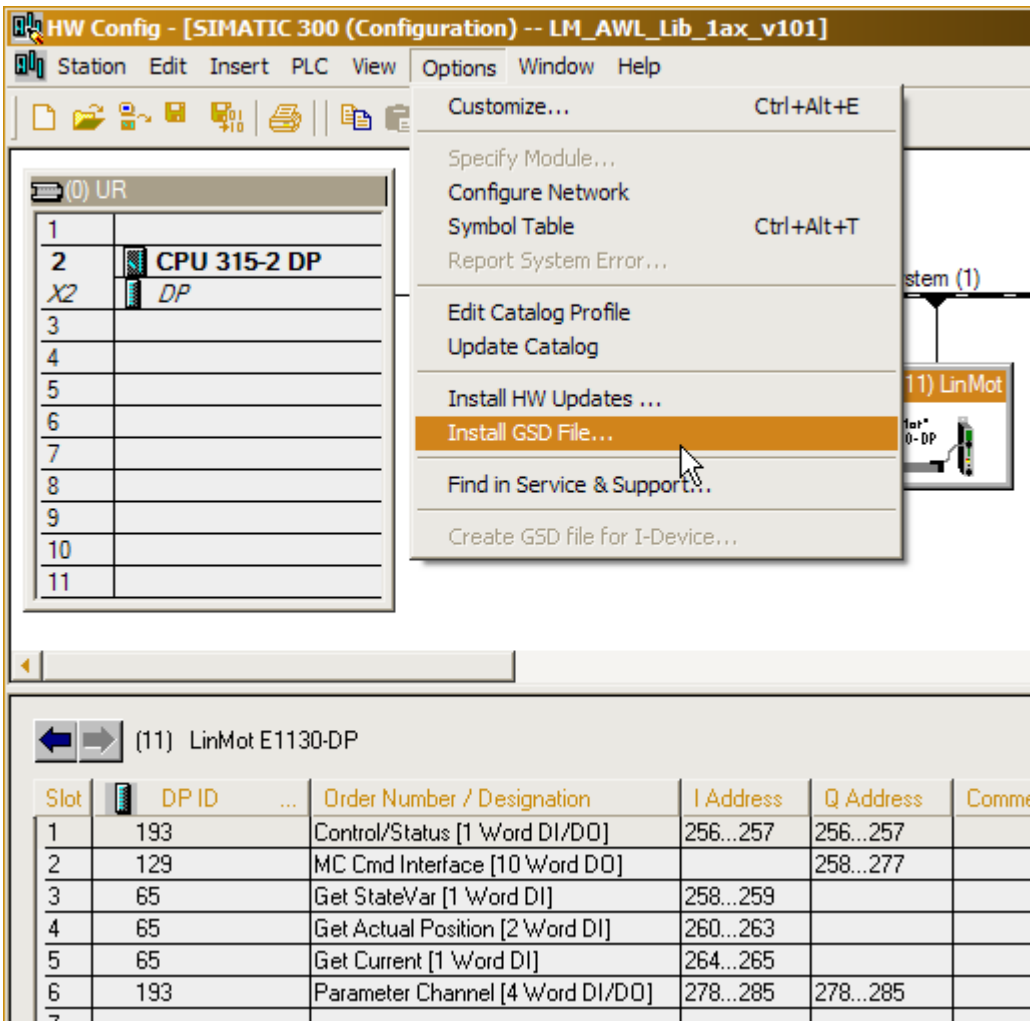


Figure 1: HW Config

Note regarding slot 6

The module “Parameter Channel [4 Word DI/DO]” is optional and is only required when using the config function blocks (Lmcf_...).

1.2 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.

The only setting that must be done on the controller according to the Profibus interface is setting 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.

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.

2. Data Types

2.1 Axis communication

2.1.1 tstLM_Axis (UDT100)

The struct tstLM_Axis contains all data required for a proper communication between the function blocks and the controller.

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	ComData_IN	STRUCT		Data to be read from bus
+0.0	StatusWord	WORD	W#16#0	StatusWord -> user manual Motion Cor
+2.0	StateVar	WORD	W#16#0	StateVar -> user manual Motion Cont
+4.0	ActualPositionCom	DWORD	DW#16#0	Actual Motor Position (not scaled /
+8.0	ActualCurrentCom	WORD	W#16#0	Actual Motor Current (not scaled /
=10.0		END_STRUCT		
+10.0	ComData_OUT	STRUCT		Data to be written on bus
+0.0	ControlWord	STRUCT		Control Word -> user manual Motion C
+0.0	CW_JogMovePlus	BOOL	FALSE	
+0.1	CW_JogMoveMinus	BOOL	FALSE	
+0.2	CW_Reserved	BOOL	FALSE	
+0.3	CW_Home	BOOL	FALSE	
+0.4	CW_ClearanceCheck	BOOL	FALSE	
+0.5	CW_GoToInitialPosition	BOOL	FALSE	
+0.6	CW_Linearizing	BOOL	FALSE	
+0.7	CW_PhaseSearch	BOOL	FALSE	
+1.0	CW_SwitchOn	BOOL	FALSE	
+1.1	CW_SVE	BOOL	FALSE	
+1.2	CW_nQuickStop	BOOL	FALSE	
+1.3	CW_EnableOperation	BOOL	FALSE	
+1.4	CW_nAbort	BOOL	FALSE	
+1.5	CW_nFreeze	BOOL	FALSE	
+1.6	CW_GoToPosition	BOOL	FALSE	
+1.7	CW_ErrorAcknowledge	BOOL	FALSE	
=2.0		END_STRUCT		
+2.0	MCCommand	STRUCT		Motion Command -> user manual Motior
+0.0	MCHeader	STRUCT		Motion Command Header
+0.0	MasterID	BYTE	B#16#0	Motion Command Master ID
+1.0	SubID	BYTE	B#16#0	Motion Command Sub ID
=2.0		END_STRUCT		
+2.0	MCParaWord0	WORD	W#16#0	Parameter Word 0
+4.0	MCParaWord1	WORD	W#16#0	Parameter Word 1
+6.0	MCParaWord2	WORD	W#16#0	Parameter Word 2
+8.0	MCParaWord3	WORD	W#16#0	Parameter Word 3
+10.0	MCParaWord4	WORD	W#16#0	Parameter Word 4
+12.0	MCParaWord5	WORD	W#16#0	Parameter Word 5
+14.0	MCParaWord6	WORD	W#16#0	Parameter Word 6
+16.0	MCParaWord7	WORD	W#16#0	Parameter Word 7
+18.0	MCParaWord8	WORD	W#16#0	Parameter Word 8
=20.0		END_STRUCT		
=22.0		END_STRUCT		
+32.0	ConfigData_IN	STRUCT		Config data to be read from bus -> u
+0.0	Status	WORD	W#16#0	Config Status Word
+2.0	ArgumentW0	WORD	W#16#0	Config Argument Word 0 in
+4.0	ArgumentW1	WORD	W#16#0	Config Argument Word 1 in
+6.0	ArgumentW2	WORD	W#16#0	Config Argument Word 2 in
=8.0		END_STRUCT		
+40.0	ConfigData_OUT	STRUCT		Data to be written on bus -> user ma
+0.0	Control	WORD	W#16#0	Config Control Word
+2.0	ArgumentW0	WORD	W#16#0	Config Argument Word 0 out
+4.0	ArgumentW1	WORD	W#16#0	Config Argument Word 1 out
+6.0	ArgumentW2	WORD	W#16#0	Config Argument Word 2 out
=8.0		END_STRUCT		
+48.0	AxisState	INT	0	
+50.0	CommandRunning	BOOL	FALSE	Axis has command running
+50.1	CommandAborted	BOOL	FALSE	Running command has been aborted
+50.2	ConfigChannelBusy	BOOL	FALSE	Config channel is busy
=52.0		END_STRUCT		

Figure 2: tstLM_Axis

Note

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

2.2 Data Types of the Config Function Blocks

2.2.1 tstLM_CfgUPIDListEntry (UDT101)

The struct tstLM_CfgUPIDListEntry contains the number and the value of a LinMot parameter.
Can be used in a DB with the following config function blocks:

- LMcf_GetModUPIDList
- LMcf_WriteUPIDList

Adresse	Name	Typ	Anfangswert
0.0		STRUCT	
+0.0	UPID	WORD	W#16#0
+2.0	UPIDValue	DWORD	DW#16#0
=6.0		END_STRUCT	

Figure 3: tstLM_CfgUPIDListEntry

2.2.2 tstLM_CfgCTEntry (UDT102)

The struct tstLM_CfgCTEntry contains all data for one entry line of the command table on the controller.
Used in the following config function block:

- LMcf_CTAccess

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	CTEntryWord0	WORD	W#16#0	Command entry version ID fix A701h
+2.0	CTEntryWord1	WORD	W#16#0	Linked Command Entry ID (ID=FFFFh means not linked)
+4.0	CTEntryWord2	WORD	W#16#0	Motion Command Header
+6.0	CTEntryWord3	WORD	W#16#0	Motion Command Parameters (Byte 0..1)
+8.0	CTEntryWord4	WORD	W#16#0	Motion Command Parameters (Byte 2..3)
+10.0	CTEntryWord5	WORD	W#16#0	Motion Command Parameters (Byte 4..5)
+12.0	CTEntryWord6	WORD	W#16#0	Motion Command Parameters (Byte 6..7)
+14.0	CTEntryWord7	WORD	W#16#0	Motion Command Parameters (Byte 8..9)
+16.0	CTEntryWord8	WORD	W#16#0	Motion Command Parameters (Byte 10..11)
+18.0	CTEntryWord9	WORD	W#16#0	Motion Command Parameters (Byte 12..13)
+20.0	CTEntryWord10	WORD	W#16#0	Motion Command Parameters (Byte 14..15)
+22.0	CTEntryWord11	WORD	W#16#0	Motion Command Parameters (Byte 16..17)
+24.0	CTEntryWord12	WORD	W#16#0	Motion Command Parameters (Byte 18..19)
+26.0	CTEntryWord13	WORD	W#16#0	Motion Command Parameters (Byte 20..21)
+28.0	CTEntryWord14	WORD	W#16#0	Motion Command Parameters (Byte 22..23)
+30.0	CTEntryWord15	WORD	W#16#0	Motion Command Parameters (Byte 24..25)
+32.0	CTEntryWord16	WORD	W#16#0	Motion Command Parameters (Byte 26..27)
+34.0	CTEntryWord17	WORD	W#16#0	Motion Command Parameters (Byte 28..29)
+36.0	CTEntryWord18	WORD	W#16#0	Motion Command Parameters (Byte 30..31)
+38.0	CTEntryWord19	WORD	W#16#0	Entry Name Character (Byte 0..1)
+40.0	CTEntryWord20	WORD	W#16#0	Entry Name Character (Byte 2..3)
+42.0	CTEntryWord21	WORD	W#16#0	Entry Name Character (Byte 4..5)
+44.0	CTEntryWord22	WORD	W#16#0	Entry Name Character (Byte 6..7)
+46.0	CTEntryWord23	WORD	W#16#0	Entry Name Character (Byte 8..9)
+48.0	CTEntryWord24	WORD	W#16#0	Entry Name Character (Byte 10..11)
+50.0	CTEntryWord25	WORD	W#16#0	Entry Name Character (Byte 12..13)
+52.0	CTEntryWord26	WORD	W#16#0	Entry Name Character (Byte 14..15)
+54.0	CTEntryWord27	WORD	W#16#0	Reserved for future use
+56.0	CTEntryWord28	WORD	W#16#0	Reserved for future use
+58.0	CTEntryWord29	WORD	W#16#0	Reserved for future use
+60.0	CTEntryWord30	WORD	W#16#0	Reserved for future use
+62.0	CTEntryWord31	WORD	W#16#0	Reserved for future use
=64.0		END_STRUCT		

Figure 4: tstLM_CfgCTEntry

Note

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

2.2.3 tstLM_CfgCTPresenceList (UDT103)

The struct tstLM_CfgCTPresenceList contains bit coded what lines of the command table have a command defined (0 = entry defined, 1 = entry not defined).

Used in the following config function block:

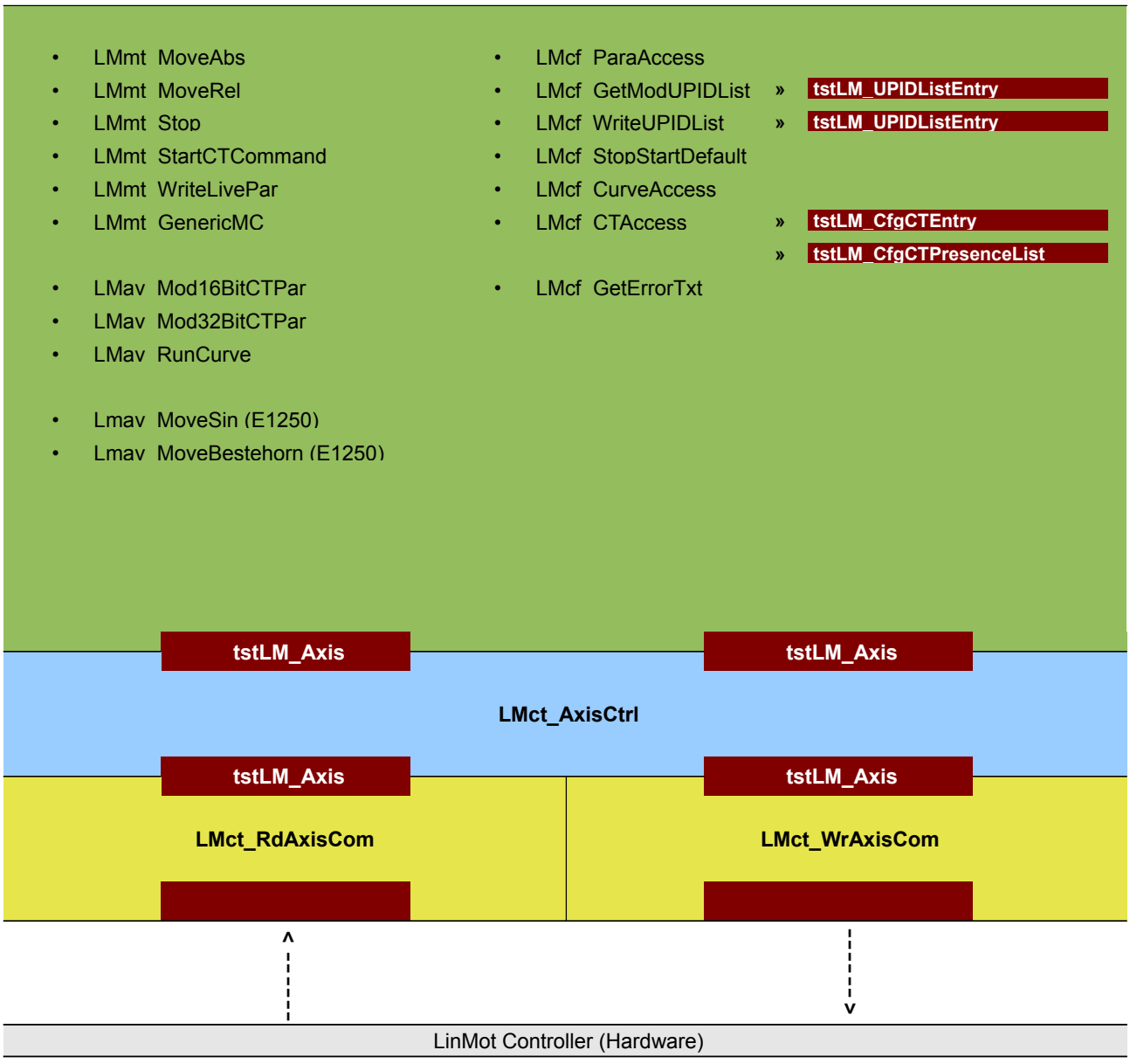
- Lmcf_CTAccess

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	CTPList0	DWORD	DW#16#0	Entries 0..31
+4.0	CTPList1	DWORD	DW#16#0	Entries 32..63
+8.0	CTPList2	DWORD	DW#16#0	Entries 64..95
+12.0	CTPList3	DWORD	DW#16#0	Entries 96..127
+16.0	CTPList4	DWORD	DW#16#0	Entries 128..159
+20.0	CTPList5	DWORD	DW#16#0	Entries 160..191
+24.0	CTPList6	DWORD	DW#16#0	Entries 192..223
+28.0	CTPList7	DWORD	DW#16#0	Entries 224..255
=32.0		END_STRUCT		

Figure 5: tstLM_CfgCTPresenceList

3. Function Blocks

3.1 Overview and dependencies



3.2 IO and Axis Control

3.2.1 LMct_RdAxisCom (FB120)

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

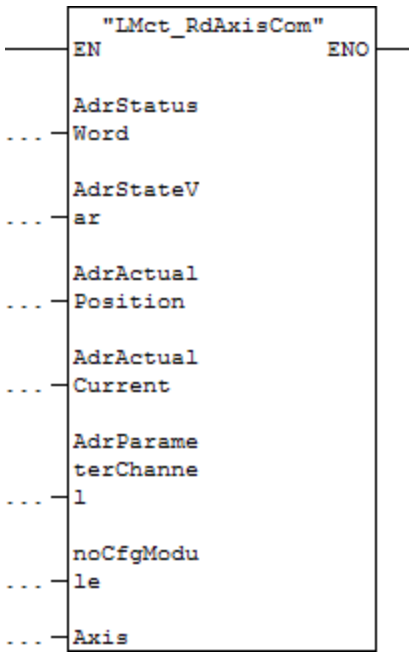


Figure 6: LMct_RdAxisCom

Inputs		
Name	**Data type**	**Description**
AdrStatusWord	Int	Input address of *Control/StatusWord* module
AdrStateVar	Int	Input address of *Get StateVar* module
AdrActualPosition	Int	Input address of *Get Actual Position* module
AdrActualCurrent	Int	Input address of *Get Current* module
AdrParameterChannel	Int	Input address of *Parameter Channel* module
NoCfgModule	Bool	Set to TRUE if the *Parameter Channel* module is **NOT** used
Axis	tstLM_Axis	Axis reference (IN_OUT)

3.2.2 LMct_WrAxisCom (FB121)

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

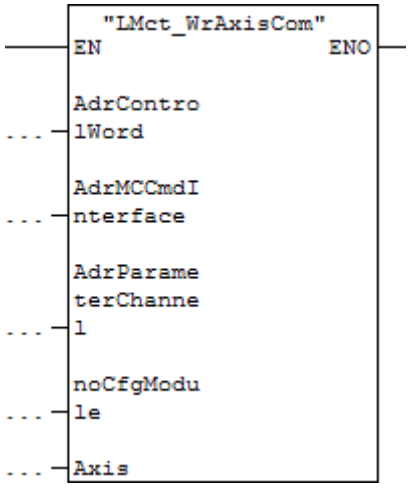


Figure 7: LMct_WrAxisCom

Inputs		
Name	**Data type**	**Description**
AdrControlWord	Int	Output address of *Control/StatusWord* module
AdrMCCmdInterface	Int	Output address of *MC CMD Interface* module
AdrParameterChannel	Int	Output address of *Parameter Channel* module
NoCfgModul	Bool	Set to TRUE if the *Parameter Channel* module is **NOT** used
Axis	tstLM_Axis	Axis reference (IN_OUT)

3.2.3 LMct_AxisCtrl (FB122)

This function block controls the state machine of a LinMot controller. The outputs show the status of the axis.

All IO and axis control function blocks (chapter 3.2.1 – 3.2.3) must be called cyclically!

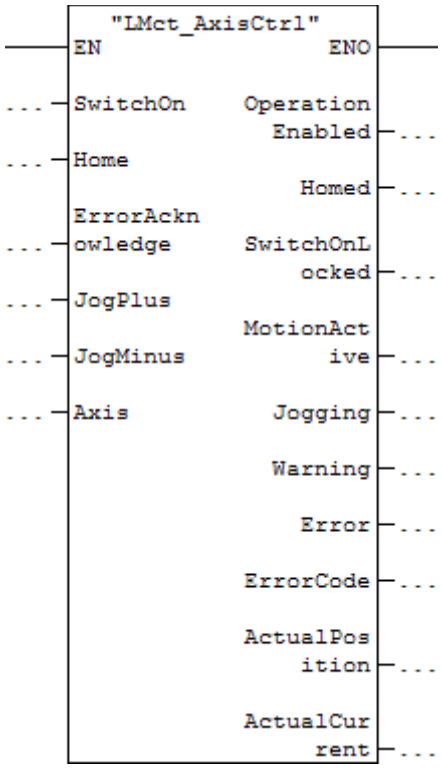


Figure 8: LMct_AxisCtrl

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

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	Int	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)

3.3 MC Function Blocks

3.3.1 LMmt_MoveAbs (FB123)

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

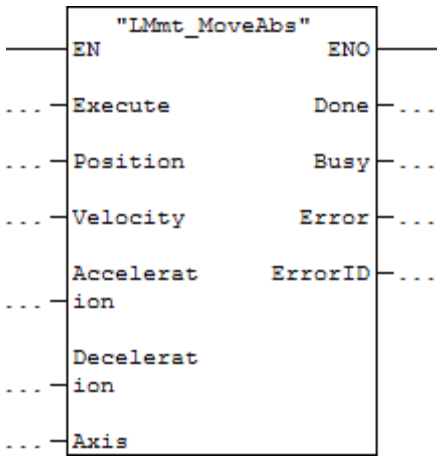


Figure 9: LMmt_MoveAbs

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

Table 1: 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	Int	ErrorID (See chapter 4. Error Descriptions)

Table 2: Outputs LMmt_MoveAbs

3.3.2 LMmt_MoveRel (FB124)

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

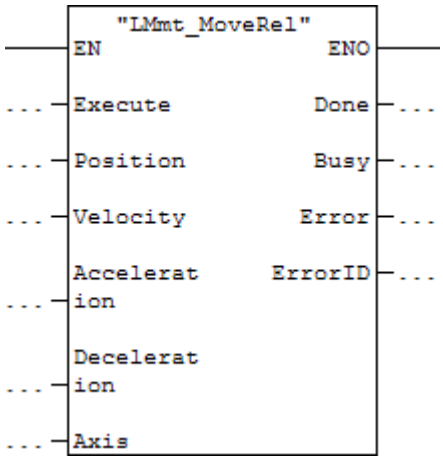


Figure 10: LMmt_MoveRel

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

Table 3: 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	Int	ErrorID (See chapter 4. Error Descriptions)

Table 4: Outputs LMmt_MoveRel

3.3.3 LMmt_StartCTCommand (FB125)

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

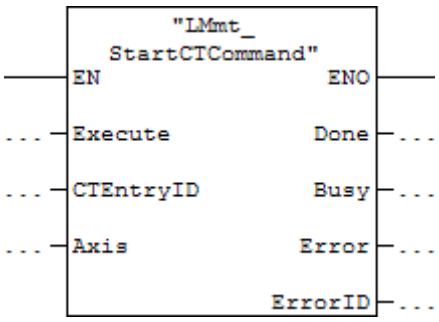


Figure 11: LMmt_StartCTCommand

Inputs			
Name	**Data type**	**Range**	**Description**
Execute	Bool		Execute command (rising edge)
CTEntryID	Int	1...255	ID of the line of the Command Table
Axis	tstLM_Axis		Axis reference (IN_OUT)

Table 5: Inputs LMmt_StartCTCommand

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

Table 6: Outputs LMmt_StartCTCommand

3.3.4 LMmt_Stop (FB126)

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

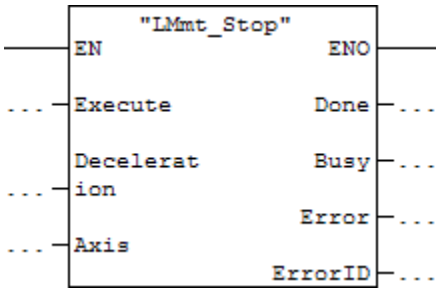


Figure 12: LMmt_Stop

Inputs			
Name	**Data type**	**Range**	**Description**
Execute	Bool		Execute command (rising edge)
Deceleration	Real		Deceleration in [m/s^2]
Axis	tstLM_Axis		Axis reference (IN_OUT)

Table 7: 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	Int	ErrorID (See chapter 4. Error Descriptions)

Table 8: Outputs LMmt_Stop

3.3.5 LMmt_WriteLivePar (FB127)

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

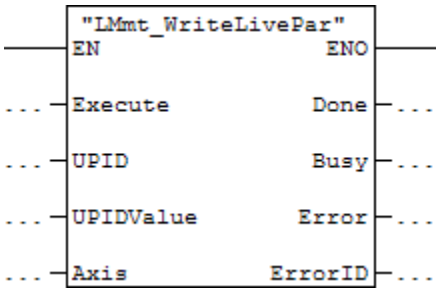


Figure 13: LMmt_WriteLivePar

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

Table 9: Inputs LMmt_WriteLivePar

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

Table 10: Outputs LMmt_WriteLivePar

Note

For more advanced parameter access use the function block LMcf_ParaAccess.

3.3.6 LMmt_GenericMC (FB128)

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

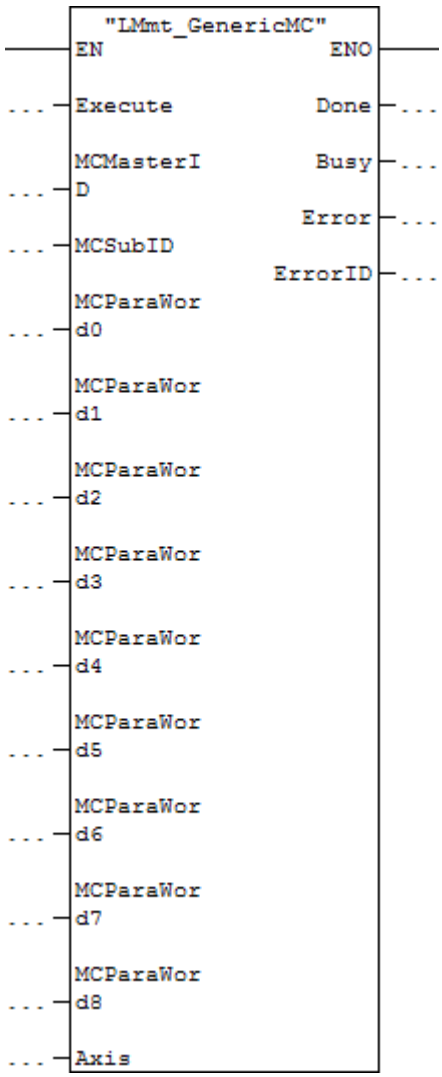


Figure 14: LMmt_GenericMC

Inputs			
Name	**Data type**	**Range**	**Description**
Execute	Bool		Execute command (rising edge)
MCMasterID	Byte		Motion Command Master ID
MCSubID	Byte		Motion Command Sub ID
MCParaWord0	Word		0. Parameter word
MCParaWord1	Word		1. Parameter word
MCParaWord2	Word		2. Parameter word
MCParaWord3	Word		3. Parameter word
MCParaWord4	Word		4. Parameter word
MCParaWord5	Word		5. Parameter word
MCParaWord6	Word		6. Parameter word
MCParaWord7	Word		7. Parameter word
MCParaWord8	Word		8. Parameter word
Axis	tstLM_Axis		Axis reference (IN_OUT)

Table 11: Inputs LMmt_GenericMC

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

Table 12: Outputs LMmt_GenericMC

3.4 MC Function Blocks (Advanced)

3.4.1 LMav_Mod16BitCTPar (FB140)

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

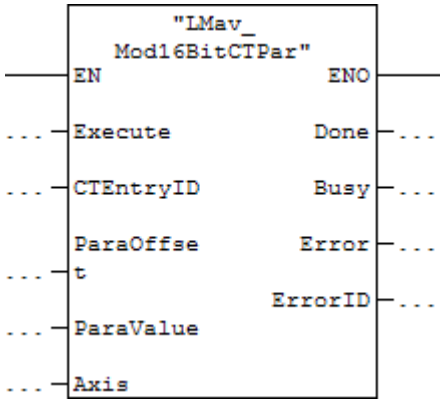


Figure 15: LMav_Mod16BitCTPar

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

Table 13: Inputs LMav_Mod16BitCTPar

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

Table 14: 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 (FB141)

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

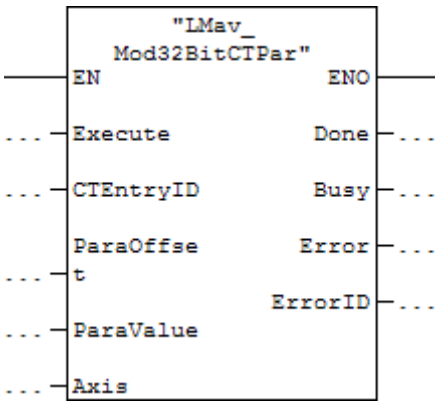


Figure 16: LMav_Mod32BitCTPar

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

Table 15: Inputs LMav_Mod32BitCTPar

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

Table 16: 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 (FB142)

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

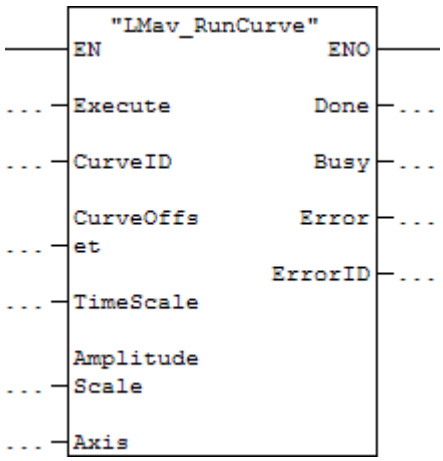


Figure 17: LMav_RunCurve

Inputs			
Name	**Data type**	**Range**	**Description**
Execute	Bool		Execute command (rising edge)
CurveID	Int	1...99	Curve number (ID)
CurveOffset	Real		Offset of the curve
TimeScale	Real	0.0...200.0	Time scale in [%]
AmplitudeScale	Real	-2000.00...+2000.00	Amplitude scale in [%]
Axis	tstLM_Axis		Axis reference (IN_OUT)

Table 17: 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	Int	ErrorID (See chapter 4. Error Descriptions)

Table 18: 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 (Advanced, E1230-DP-UC Controller)

3.5.1 LMav_MoveBestehorn_E1230 (FB145)

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

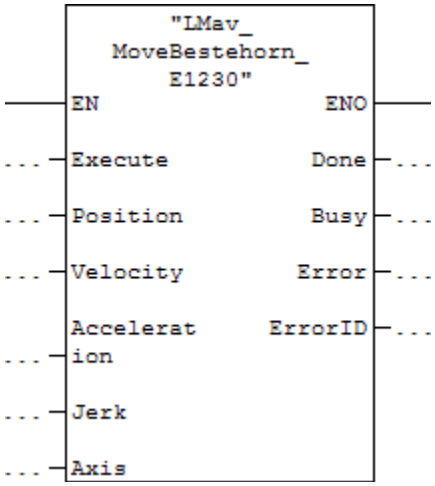


Figure 18: LMav_MoveBestehorn_E1230

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

Table 19: Inputs LMav_MoveBestehorn_E1230

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	Int	ErrorID (See chapter 4. Error Descriptions)

Table 20: Outputs LMav_MoveBestehorn_E1230

Note

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

3.5.2 LMav_MoveSin_E1230 (FB146)

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

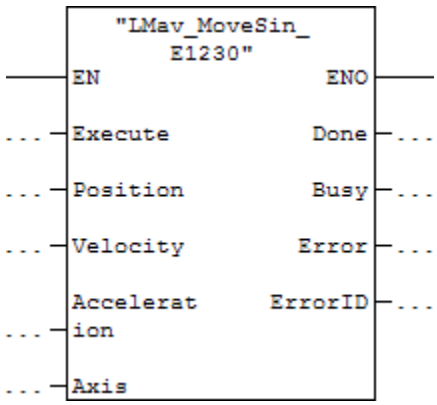


Figure 19: LMav_MoveSin_E1320

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

Table 21: Inputs LMav_MoveSin_E1230

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_MoveSin_E1320

Note

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

3.6 Config Function Blocks (E1230-DP-UC, 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, restoring the parameters of each firmware layer to default values, or getting the error text as STRING.

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

- 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.

This is required for the following function blocks and modes:

- LMcf_CTAccess Mode 0
- LMcf_CurveAccess Mode 0

3.6.1 LMcf_ParaAccess (FB200)

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.

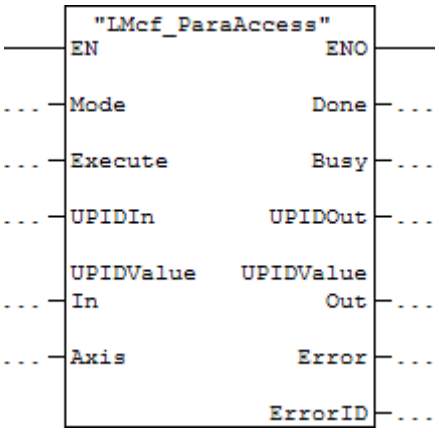


Figure 20: LMcf_ParaAccess

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

Table 23: Inputs LMcf_ParaAccess

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

Table 24: 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 25: Modes of LMcf_ParaAccess

3.6.2 LMcf_GetModUPIDList (FB201)

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

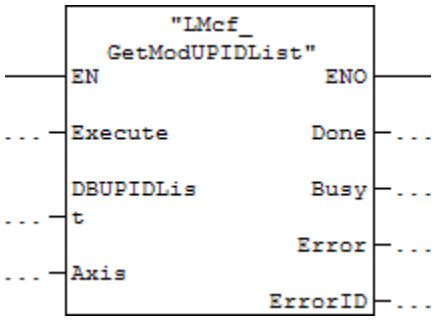


Figure 21: LMcf_GetModUPIDList

Inputs		
Name	**Data type**	**Description**
Execute	Bool	Execute command (rising edge)
DBUPIDList	Block_DB	DB where the parameters are stored
Axis	tstLM_Axis	Axis reference (IN_OUT)

Table 26: 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	Int	ErrorID (See chapter 4. Error Descriptions)

Table 27: Outputs LMcf_GetModUPIDList

Attention

The size of the DB connected to *DBAddressData* has to be big enough to store the curve!
An example DB is part of the S7 example project coming with this document.

3.6.3 LMcf_WriteUPIDList (FB202)

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

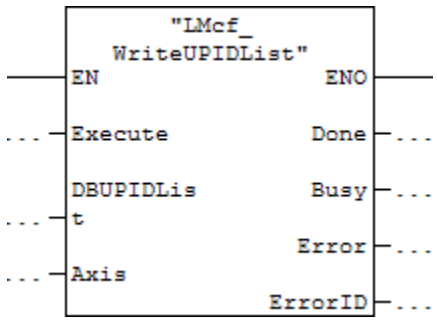


Figure 22: LMcf_WriteUPIDList

Inputs		
Name	**Data type**	**Description**
Execute	Bool	Execute command (rising edge)
DBUPIDList	Block_DB	DB where the parameters are stored
Axis	tstLM_Axis	Axis reference (IN_OUT)

Table 28: Inputs LMcf_WriteUPIDList

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

Table 29: Outputs LMcf_WriteUPIDList

Attention

The function block stops writing as soon as it finds a UPID number in the array that is zero. Therefore a parameter with UPID = 0 must follow the last valid parameter in the DB where the entries are stored.

3.6.4 LMcf_StopStartDefault (FB203)

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.

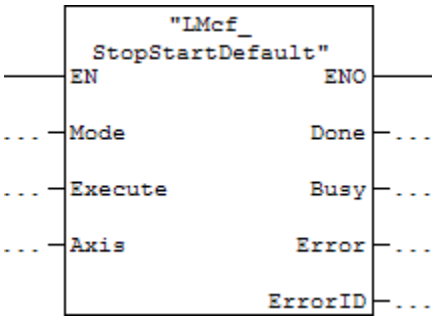


Figure 23: LMcf_StopStartDefault

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

Table 30: Inputs LMcf_StopStartDefault

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

Table 31: 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 32: 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!**

3.6.5 LMcf_CurveAccess (FB204)

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.

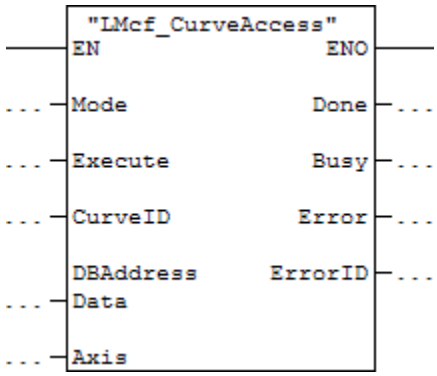


Figure 24: LMcf_CurveAccess

Inputs			
Name	**Data type**	**Range**	**Description**
Mode	Int	0...4	Mode
Execute	Bool		Execute command (rising edge)
CurveID	Int	1...99	Curve number (ID)
DBAddressData	Block_DB		Data Block with/for curve data
Axis	tstLM_Axis		Axis reference (IN_OUT)

Table 33: Inputs LMcf_CurveAccess

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

Table 34: 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	DBAddressData	Add Curve (RAM)
3	DBAddressData	Modify Curve (RAM)
4	CurveID, DBAddressData	Get Curve (data is stored in DB at DBAddressData)

Table 35: Modes of LMcf_CurveAccess

Attention

The size of the DB connected to *DBAddressData* has to be big enough to store the curve!
An example DB is part of the S7 example project coming with this document.

3.6.6 LMcf_CTAccess (FB205)

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.

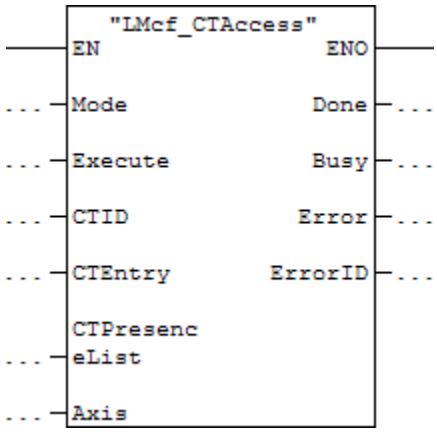


Figure 25: LMcf_CTAccess

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

Table 36: Inputs LMcf_CTAccess

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

Table 37: 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 38: Modes of LMcf_CTAccess

3.6.7 LMcf_GetErrorTxt (FB206)

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

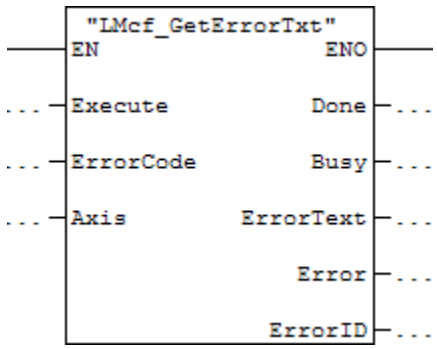


Figure 26: LMcf_GetErrorTxt

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

Table 39: Inputs LMcf_GetErrorTxt

Outputs		
Name	**Data type**	**Description**
Done	Bool	Command executed / Error text read
Busy	Bool	Command active
ErrorText	String[32]	Error text as STRING
Error	Bool	Error in function block
ErrorID	Int	ErrorID (See chapter 4. Error Descriptions)

Table 40: Outputs LMcf_GetErrorTxt

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 41: 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
06h	DB size to small	Connected DB is to small
C0h	UPID error	Unknown UPID selected, check UPID input
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 42: Error ID's of the Config Function Blocks

5. Example Project

There is an example project included in the library package that shows the general integration to a SIMATIC 300 PLC.

In the project all function blocks of the library are included.
But only the following are called in FB100 (Axis_A):

- FB120 LMct_RdAxisCom Read axis data from bus
- FB121 LMct_WrAxisCom Write axis data to bus
- FB122 LMct_AxisCtrl Axis control function block
- FB123 LMmt_MoveAbs Move absolute
- FB124 LMmt_MoveRel Move relative
- FB200 LMcf_ParaAccess Access parameters of the controller

Additionally to FB100 (DB100) in OB1 the two functions FC1 (CycleA) and FC2 (AxisA_Init) are called.
FC1 is an example of how to move between four positions. FC2 shows the initialisation and error acknowledge of the axis.

Included as well is a variable table (AxisA_IO) to control the above mentioned functions and 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…