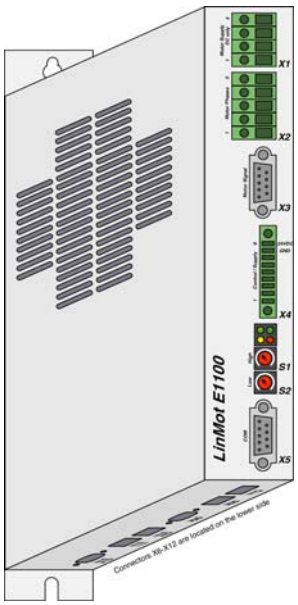
LinMot®

CANopen

Documentation of the CANopen Interface of the following Controllers:

- **E1100-CO**
- **E1100-CO-HC**
- **E1100-GP (with CANopen Firmware loaded)**
- **E1100-GP-HC (with CANopen Firmware loaded)**



CANopen Interface 3.6

User manual

NTI AG
LinMot®
Haerdlistrasse 15
CH-8957 Spreitenbach

Tel.: +41 (0)56 419 9191
Fax: +41 (0)56 419 9192
Email: office@linmot.com
Internet: www.linmot.com

Document version 3.6 / FM, Mai, 2006

1. System overview

The LinMot CANopen controllers E1100-CO support the Communication Profile CiA DS301.
Further information on CANopen can be found under: http://www.can-cia.de/

The following resources are available:
- 3 T_PDO
- 3 R_PDO
- 1 T_SDO
- 1 R_SDO

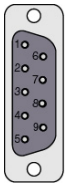
The supported protocols include:
- NMT Error Control (Nodeguarding Protocol or HeartBeat Protocol)
- PDO (Transmission type 254 and 1)
- SDO Upload and Download
- NMT (Start, Stop, Enter PreOp, Reset Node, Reset Communication)
- Boot-Up Message

The baud rate can be selected by parameter or direct by BTR register.

2. Connecting to the CAN bus

Pinout of the COM Connector:

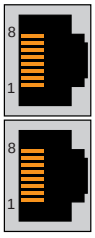
DSBU 9 male:



Pin 1	RS-485 Y	Pin 6	RS-485 B
Pin 2	RS-232 TX	Pin 7	RS-485 Z
Pin 3	RS-232 RX	**Pin 8**	**CAN L**
Pin 4	RS-485 A	**Pin 9**	**CAN H**
Pin 5	**GND**		

Pinout of the CMD Connector:

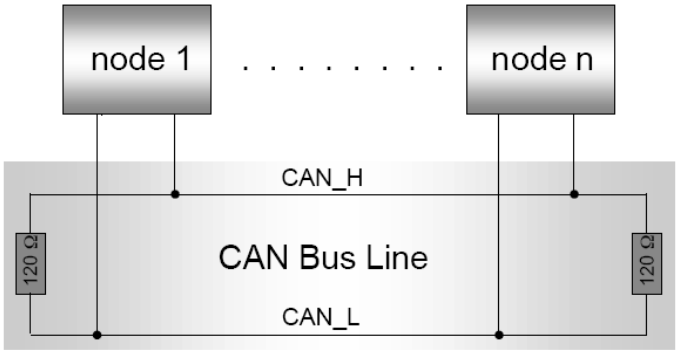
2xRJ45 with 1:1 connected signals. Standard twisted pairs: 1/2, 3/6, 4/5, 7/8.
Ethernet cables according standard



Pin 1	RS485 A
Pin 2	RS485 B
Pin 3	RS485 Y
Pin 4/5	**Ground**
Pin 6	RS485 Z
Pin 7	**CAN H**
Pin 8	**CAN L**

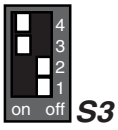
CAN Termination

The CANbus must be terminated by two 120 Ohm resistors at both ends of the bus line, according the following scheme:



For easy installation, the LinMot CANopen controller has built in termination resistors, which can be activated, if the LinMot controller is at the end of the bus line, and if there is no termination in the connector.

S3
ON – OFF
Interface
CAN Term
RS485 Term
RS485/232



S3

The built in termination resistor for the CAN bus can be activated by setting the dip switch “CAN Term” to “ON” If the dipswitch S3.4 Interface is set to “OFF”, the CANopen Interface is deactivated.

3. CANopen Parameters

The CANopen Servo Controllers have an additional parameter tree branch, which can be configured with the distributed LinMot Commander software. With these parameters, the CANopen behaviour can be configured. The software LinMot Commander can be downloaded from http://www.linmot.com under the section download, software & manuals.

Dis-/Enable

With the Dis-/Enable parameter the LinMot servo controller can be run without the CANopen going online. So in first step the system can be configured and run without any bus connection.

CANopen Interface\ Dis-/Enable	
Disable	Servo controller runs without CANopen.
Enable	Servo controller runs only with a CANopen connection.

IMPORTANT: To activate the CANopen Interface, the Dip-Switch “Interface” at the bottom of the drive has to be set to “ON”

Baud Rate In this section the parameters for the baud rate selection are located.

Baud Rate Source Select

Defines the source of the baud rate definition.

CANopen Interface\ Baud Rate \Baud Rate Source Select	
By Hex Switch S1	CAN bus baud rate dependent on S1: 0 = By BTR 1 = 125 kBit/s 2 = 250 kBit/s 3 = 500 kBit/s 4 = 1 Mbit/s
By Parameter	The CAN bus baud rate is selected by the "Baudrate Parameter": - 125 kBit/s [1] - 250 kBit/s [2] - 500 kBit/s [3] - 1 Mbit/s [4]
By BTR	CAN bus baud rate is according the Bit Timing Register

Baud Rate BTR Value

For special applications, where no standard setting for the baud rate works, this parameter defines the bit timing for the CAN bus. The setting of the baud rate by Bit Timing Register is only necessary on special bus configurations: For example, if there are devices on the bus that have slow optocouplers. (The settings of the BTR are according the documentation of Infineon C167CR running at 40MHz).

Baud Rate Parameter Definition

The Baud rate parameter defines the CAN bus baud rate for the CANopen connection.

CANopen Interface\ Baud Rate\ Baud Rate Parameter Definition	
125 kBit/s	CAN bus baud rate = 125 kBit/s
250 kBit/s	CAN bus baud rate = 250 kBit/s
500 kBit/s	CAN bus baud rate = 500 kBit/s
1 Mbit/s	CAN bus baud rate = 1 Mbit/s

MACID In this section the MACID (controller number) can be configured

MACID Source Select

The MACID parameter defines the source of the MACID (Node Address).

CANopen Interface\ MACID\ MACID Source Select	
By Hex Switch S2	The MACID is determined by the Hex Switch S2
By Hex Switches S1 and S2	The MACID is determined by the two Hex Switches S1 and S2
By Parameter	The MACID is determined by parameter setting
Parameter Value	The MACID, when “Parameter” is selected

With Default Settings, the MAC-ID and the baud rate are selected by the two rotary Hex Switches S1 and S2

MACID Parameter Value

The ID, when “By Parameter” is selected as source.

PDO Mapping

TxPDO 1 These parameters define the mapping of the transmit PDO1. There can be mapped 4 Words in total.

CANopen Interface\ PDO Mapping\ TxPDO 1	
Status Word [1W]	If this boolean parameter is set, the Status Word is transmitted with TxPDO1 (see variable 1D51h).
State Var [1W]	If this boolean parameter is set, the State Var (High Byte = State Nr. / Low Byte = SubState) is transmitted with TxPDO1 (see variable 1B62h).
Logged Error Code [1W]	If this boolean parameter is set, the Logged Error Code is transmitted with TxPDO1 (see variable 1D96h).
Warn Word [1W]	If this boolean parameter is set, the Warn Word (= bit coded warnings) is transmitted with TxPDO1 (see variable 1D8Eh).
Demand Current [1W]	If this boolean parameter is set, the Demand Current value (= motor current) is transmitted with TxPDO1 (see variable 1B93h).
Actual Position low word [1W]	If this boolean parameter is set, then the lower 16 bit of the Actual Position (32 bit value, see variable 1B8Dh) is transmitted with TxPDO1.
Actual Position high word [1W]	If this boolean parameter is set, then the higher 16 bit of the Actual Position (32 bit value, see variable 1B8Dh) is transmitted with TxPDO1.
By UPID	This parameter can be used for free mapping of any parameter or variable to TxPDO1 (mapping through Unique Parameter ID = UPID, 0 = no mapping). The corresponding data size in TxPDO1 is either 1 Word, if parameter or variable type is 16 Bit or less, or 2 Words, if the type is 32 Bit.

TxPDO 2 These parameters define the mapping of the transmit PDO 2. There can be mapped 4 Words in total.

CANopen Interface\ PDO Mapping\ TxPDO 2	
Motion Cmd Status [1W]	Feedback of the Motion Command Header (Toggle, etc?)
Actual Position 16 Bit [1W]	If this boolean parameter is set, the actual motor position in 16 bit format is transmitted with TxPDO2 (see variable 1B95h).
Demand Current [1W]	If this boolean parameter is set, the Demand Current value (= motor current) is transmitted with TxPDO2 (see variable 1B93h).
Demand Position 16 Bit [1W]	If this boolean parameter is set, the Demand Position in 16 bit format is transmitted with TxPDO2 (position setpoint, see variable 1B94h).
By UPID	This parameter can be used for free mapping of any parameter or variable to TxPDO2 (mapping through Unique Parameter ID = UPID, 0 = no mapping). The corresponding data size in TxPDO2 is either 1 Word, if parameter or variable type is 16 Bit or less, or 2 Words, if the type is 32 Bit.

TxPDO 3 These parameters define the mapping of the transmit PDO 3. There can be mapped 4 Words in total.

CANopen Interface\ PDO Mapping\ TxPDO 3	
Dummy	Reserved for further use.

RxPDO 1 These parameters define the mapping of the receive PDO1. There can be mapped 4 Words in total.

CANopen Interface\ PDO Mapping\ RxPDO 1	
Control Word [1W]	If this boolean parameter is set, then the Control Word has to be transmitted with RxPDO1 (see variable 1D4Ch)
Motion Cmd Header + Par Byte 0..3 [3W]	Motion Command Interface (Header and the first 4 bytes of the command parameters).
By UPID	For free mapping, every parameter or variable can be mapped by its UPID (Unique Parameter ID). The Size is either 1 Word, if type is 16 Bit or less, or 2 Words, if the type is 32 Bit.

RxPDO 2 These parameters define the mapping of the receive PDO2. There can be mapped 4 Words in total.

CANopen Interface\ PDO Mapping\ RxPDO 2	
Motion Cmd Header + Par Byte 0..5 [4W]	Motion Command Interface (Header and the first 6 bytes of the command parameters)
By UPID	For free mapping, every parameter or variable can be mapped by its UPID (Unique Parameter ID). The Size is either 1 Word, if type is 16 Bit or less, or 2 Words, if the type is 32 Bit.

RxPDO 3 These parameters define the mapping of the receive PDO3. There can be mapped 4 Words in total.

CANopen Interface\ PDO Mapping\ RxPDO 3	
CMD Slave Header + Par Byte 6..7 [2W]	Command Interface (Slave Header and Byte 6..7 of the Parameters)
CMD Slave Header + Par Byte 6..9 [3W]	Command Interface (Slave Header and Byte 6..9 of the Parameters)
CMD Slave Header + Par Byte 6..11 [4W]	Command Interface (Slave Header and Byte 6..11 of the Parameters)
Direct Par X [1W]	Direct Parameter Channel for setting live parameters during runtime (only 16 bit parameters).
Direct Par X UPID	UPID (**U**nique **P**arameter **ID**) of the selected Parameter
By UPID	For free mapping, every parameter or variable can be mapped by its UPID (Unique Parameter ID). The Size is either 1 Word, if type is 16 Bit or less, or 2 Words, if the type is 32 Bit.

PDO Configuration

TxPDO 1..3 These parameters define the bus parameters of the transmit PDO1..3.

TxPDO 1..3 Enable

Selector for enabling/disabling the transmit PDO1..3.

CANopen Interface\ PDO Configuration\ TxPDO 1..3\ TxPDO 1..3 Enable	
Disable	The PDO is deactivated
Enable	The PDO is activated

Transmission Type

This defines the transmission type according DS 301. Default Value is 254 (Asynchron with inhibit Time). Type 1 (cyclic synchron) is supported as well.

Inhibit Time

Defines the minimal time between two send events.

Event Time

Defines the maximal time between two send events.

RxPDO 1..3 These parameters define the bus parameters of the receive PDO1..3.

CANopen Interface\ PDO Configuration\ RxPDO 1..3	
Disable	The PDO is deactivated
Enable	The PDO is activated

NMT Error Control

Nodeguarding Protocol

Directory for configuring the nodeguarding.

Nodeguarding Enable

Enable/Disable the nodeguarding feature.

CANopen Interface\ NMT Error Control\ Nodeguarding Protocol\ Nodegarding Enable	
Disable	The Nodeguarding Protocol is deactivated.
Enable	The Nodeguarding Protocol is activated.
Guard Time	The Guard time, when Nodeguarding is activated.

Gurard Time

The Guard time, when Nodeguarding is activated.

Heartbeat Protocol

These parameters configure the Heartbeat Protocol.

CANopen Interface\ NMT Error Control\ Heartbeat Protocol	
Produce	Cyclic Heartbeat is produced.
Consume	Cyclic Heartbeat is consumed
Producer Time	Cycle Time for producing Heartbeat
Consumer Time	Guarding Time for consumed Heartbeat
Consumed Node ID (Master)	Node ID of the Master

Legacy Sync Watchdog

These parameters configure the legacy watchdog of the Sync Telegram.

Watchdog Enable

Enabling/Disabling the legacy sync watchdog feature.

CANopen Interface\ NMT Error Control\ Legacy Sync Watchdog\ Watchdog Enable	
Disable	The Sync Watchdog is deactivated.
Enable	The Sync Watchdog is activated.
Sync Cycle Time	The expected Sync Cycle Time. The Sync cycle is monitored with 1.5* Sync Cycle Time. This means that there can be configured the real expected Sync Cycle Time.

Sync Cycle Time

The expected Sync Cycle Time. The Sync cycle is monitored with 1.5* Sync Cycle Time. This means that there can be configured the real expected Sync Cycle Time.

There should only one NMT Error Control Protocol be activated.

4. Mapping of the PDO's

Mapping Table

The PDO's are default mapped according the following scheme:

Receive PDO's:

R_PDO1	R_PDO2	R_PDO3	
Control Word	CMD Header	CMD Slave Header	
	Par 1	Par 4	
	Par 2	Direct Par Channel 1	
	Par 3		

Because the CMD Interface of the LinMot Controller consists of more than 8 Bytes, it's necessary to couple two PDO together for ensuring data consistency. This is done by the "CMD Slave Header". This means, that to execute a command both headers have to be toggled. On the slave Header only the last 4 Bit are evaluated, so it's possible to simply copy the "CMD Header" from RPDO2 to the "CMD Slave Header" of RPDO3.

Transmit PDO's:

T_PDO1	T_PDO2	T_PDO3	
Status Word	CMD Status		
Run State	Actual Position		
Error Code	Actual Current		
Warn Word	Actual SetPosition		

If the application requires, the mapping can be completely changed by the PDO Mapping parameter settings. Many applications do not require to use all resources.

Default Identifier:

The default identifiers (11 Bit identifier) are allocated by the following scheme:

10	9	8	7	6	5	4	3	2	1	0
Function Code				Node ID						

This results in the following table:

Object	Function Code (binary)	COB ID (hex)	Object for Comm. Parameter / Mapping
NMT	0000	00h	- / -
SYNC	0001	80h	1005h / 1006h
Emergency	0001	81h – FFh	- / -
T_PDO1	0011	181h – 1FFh	1800h
T_PDO2	0101	281h – 2FFh	1801h
T_PDO3	0111	381h – 3FFh	1802h
R_PDO1	0100	201h – 27Fh	1400h
R_PDO2	0110	301h – 37Fh	1401h
R_PDO3	1000	401h – 47Fh	1402h
T_SDO	1011	581h – 5FFh	- / -
R_SDO	1100	601h – 67Fh	- / -

In the Pre-Operational state, this can be changed with SDO downloads by the master.

5. Motor Commands

Please refer to “Usermanual Motion Control Software”

6. State Machine

Please refer to “Usermanual Motion Control Software”

7. Interface Error Codes

Please refer to “Usermanual Motion Control Software” for the Error Codes of the MC Software. The CANopen Interface has the following additional Error Codes:

Error Code Hexadecimal	Error Description
$C1	The Controller is not compatible for CANopen
$C2	The configured ID is not valid (switches or parameter)
$C3	CANopen Error: Data out of Range
$C4	CANopen Error: Invalid Command
$C5	CANopen Error: Bus error
$C6	CANopen Error: general Bus error
$C7	CANopen Error: Bus error, stuff error
$C8	CANopen Error: Bus error, form error
$C9	CANopen Error: Bus error, ack error
$CA	CANopen Error: Bus error, bit 1 error
$CB	CANopen Error: Bus error, bit 0 error
$CC	CANopen Error: Bus error, CRC error
$CD	CANopen Error: Bus error, guard timeout
$CE	CANopen Error: Invalid UPID configured on Direct Par 1
$CF	CANopen Error: Invalid UPID configured on Direct Par 2
$D0	CANopen Error: Error: Invalid ID by Hex Switch S1
$D1	CANopen Error: Invalid Mapping in TxPDO1
$D2	CANopen Error: Invalid Mapping in TxPDO2
$D3	CANopen Error: Invalid Mapping in TxPDO3
$D4	CANopen Error: Invalid Mapping in RxPDO1
$D5	CANopen Error: Invalid Mapping in RxPDO2
$D6	CANopen Error: Invalid Mapping in RxPDO3
$D7	CANopen Error: Invalid UPID in TxPDO1 Mapping
$D8	CANopen Error: Invalid UPID in TxPDO2 Mapping
$D9	CANopen Error: Invalid UPID in TxPDO3 Mapping
$DA	CANopen Error: Invalid UPID in RxPDO1 Mapping
$DB	CANopen Error: Invalid UPID in RxPDO2 Mapping
$DC	CANopen Error: Invalid UPID in RxPDO3 Mapping

8. WarnWord

Please refer to “Usermanual Motion Control Software”

9. Object Dictionary

Index	Subindex	Description	Data Type	Value
0001h–001Fh		Data Types	DEFTYPE	
0020h		Communication Parameter	DEFSTRUCT	
	0h	Number of entries	UI8	
	1h	COB-ID	UI32	
	2h	Transmission type	UI8	
	3h	Inhibit time	UI16	
	4h	Reserved	UI8	
	5h	Event timer	UI16	
1000h		Device Type	UI32	0
1001h		Error register	UI8	
1008h		Manufacturer Device Name	Visible String	4 ASCII Zeichen, welche die 4 letzten Zahlen der Artikelnummer beinhalten
1018h		Idendity Object	Record	
	0h	Number of Entries	UI8	4
	1h	Vendor ID	UI32	0000 0156h
	2h	Product Code	UI32	
	3h	Revision Number	UI32	
	4h	Serial Number	UI32	
2000h -5FFFh		LinMot Parameters Index = 0x2000h + UPID	UI32	
	00h	Number of Entries		
	01h	RAM Value	SI32	RAM Value (rw)
	02h	ROM Value	SI32	ROM Value (rw)
	03h	Min Value	SI32	Minimal Value (ro)
	04h	Max Value	SI32	Maximal Value (ro)
	05h	Default Value	SI32	Default Value (ro)
	06h	RAM/ROM Write	SI32	RAM and ROM value can be written with the same value (wo)
	07h	Set ROM to default (OS)		Write anything to 0x2000h sub 7 to set all parameters of the OS to default values (wo). This command needs about 0.5s to finish.
	08h	Set ROM to default (MC)		Write anything to 0x2000h sub 8 to set all parameters of the MC Sw to default values (wo). This command needs about 2s to finish.
	09h	Set ROM to default (Interface)		Write anything to 0x2000h sub 9 to set all parameters of the CANopen Interface to default values (wo). This command needs about 0.5 s to finish.
	0Ah	Set ROM to default (Application)		Write anything to 0x2000h sub Ah to set all

				parameters of the Application to default values (wo)
	0Bh	Reset Controller		Write anything to 0x2000h sub Bh to reset the Controller (wo)
	20h	Start Getting UPID List		See chapter 12
	21h	Get Next UPID List item		See chapter 12
	22h	Start Getting Modified UPID List		See chapter 12
	23h	Get Next Modified UPID List item		See chapter 12
	35h	Stop MC and Application Software (for Flash access)		Write anything to 0x2000h sub 35h to stop the MC and Application SW (wo)
	36h	Start MC and Application Software		Write anything to 0x2000h sub 36h to start the MC and Application SW (wo)
	40h	Curve Service: Save to Flash		Write anything to 0x2000h sub 40h to save the curves from the RAM into the Flash ROM (wo)
	41h	Curve Service: Delete all Curves (RAM)		Write anything to 0x2000h sub 41h to Delete all Curves in the RAM (wo)
	42h	Curve Service: Poll Flash		Read anything from 0x2000h sub 42h to get the Flash state (r)
	50h	Curve Service: Add Curve		See chapter 10
	51h	Curve Service: Add Curve Info Block		See chapter 10
	52h	Curve Service: Add Curve Data		See chapter 10
	53h	Curve Service: Add Curve Data (32 Bit)		See chapter 10
	60h	Curve Service: Get Curve		See chapter 10
	61h	Curve Service: Get Curve Info Block		See chapter 10
	62h	Curve Service: Get Curve Data		See chapter 10
	70h	Get Error Log Entry Counter		See chapter 13
	71h	Get Error Log Entry Error Code		See chapter 13
	72h	Get Error Log Entry Time low		See chapter 13
	73h	Get Error Log Entry Time high		See chapter 13
	74h	Get Error Code Text Stringlet		See chapter 13
	80h	CT: Save to Flash		Write anything to 0x2000h sub 80h to save the Command Table from the RAM into the Flash ROM (w)
	80h	CT: Poll Flash		Read anything from 0x2000h sub 80h to get the Flash state (r)
	81h	CT: Delete all Entries (RAM)		Write anything to 0x2000h sub 81h to delete the complete

				Command Table in the RAM (wo)
	82h	CT: Delete Entry (Entry Nr.)		Write anything to 0x2000h + Entry Nr. Sub 82h to delete entry in the RAM
	83h	CT: Write Entry (Entry Nr.)		Write block size to 0x2000h + Entry Nr. Sub 83h to prepare entry in the RAM
	84h	CT: Write Entry Data		Write 2 Byte Data to 0x2000h + Entry Nr. Sub 84h, until block size has reached (the entry will be activated at this time)
	85h	CT: Get Entry (Entry Nr.)		Read the block size of 0x2000h + Entry Nr. Sub 85h.
	86h	CT: Get Entry Data		Read 2 byte data
	87h	CT: Get Entry List (Entry 0..31)		Read Bitfield (0=present)
	88h	CT: Get Entry List (Entry 32..63)		Read Bitfield (0=present)
	89h	CT: Get Entry List (Entry 64..95)		Read Bitfield (0=present)
	8Ah	CT: Get Entry List (Entry 96..127)		Read Bitfield (0=present)
	8Bh	CT: Get Entry List (Entry 128..159)		Read Bitfield (0=present)
	8Ch	CT: Get Entry List (Entry 160..191)		Read Bitfield (0=present)
	8Dh	CT: Get Entry List (Entry 192..223)		Read Bitfield (0=present)
	8Eh	CT: Get Entry List (Entry 224..255)		Read Bitfield (0=present)

10. Write curve into the controller via CANopen

Add curve

An curve with ID “CurveID” will be created. If a curve with the same ID already exists an error will be generated.

Index	Subindex	Data	Result
2000h + CurveID	50h	InfoBlockSize (2 bytes) + DataBlockSize (2 bytes)	00h: No error D4h: Curve already exist

Example
LinMot MACID = 1
CuveID = 1
InfoBlockSize = 70 (0046h)
DataBlockSize = 164 (00A4h)

Index = 2001h
Subindex = 50h

CAN Telegram (8 Byte Data), COB-ID 601, PLC -> LinMot Controller:

	Index		Subindex	Info Block Size		Data Block Size	
23	**01**	**20**	**50**	**46**	**00**	**A4**	**00**

Response: (8 Byte Data), COB-ID 581, LinMot Controller -> PLC:

	Index		Subindex	Unused Data		Result	
60	**01**	**20**	**50**	**00**	**00**	**00**	**00**

Add Curve Info Block

Index	Subindex	Data	Result
2000h + CurveID	51h	Unused data (2 bytes) Info Block data (2 bytes)	04h: Info Block is not finished 00h: Info Block is finished D0h: Error: Info Block was already finished

Example
Index = 2001h
Subindex = 51h
Data = 0046h

CAN Telegram (8 Byte Data), COB-ID 601, PLC -> LinMot Controller:

	Index		Subindex	Unused Data		Info Block Data	
23	**01**	**20**	**51**	**00**	**00**	**46**	**00**

Response: (8 Byte Data), COB-ID 581, LinMot Controller -> PLC:

	Index		Subindex	Unused Data		Result	
60	**01**	**20**	**51**	**00**	**00**	**04**	**00**

Add Curve Data

Index	Subindex	Data	Result
2000h + CurveID	52h	Unused data (2 bytes) Data Block data (2 bytes)	04h: Data Block is not finished 00h: Data Block is finished D0h: Error: Data Block was already finished

Example
Index = 2001h
Subindex = 52h
Data = 2710h

CAN Telegram (8 Byte Data), COB-ID 601, PLC -> LinMot Controller:

	Index		Subindex	Unused Data		Data Block Data	
23	**01**	**20**	**52**	**00**	**00**	**10**	**27**

Response: (8 Byte Data), COB-ID 581, LinMot Controller -> PLC:

	Index		Subindex	Unused Data		Result	
60	**01**	**20**	**52**	**00**	**00**	**04**	**00**

Add Curve Data 32 Bit

Index	Subindex	Data	Result
2000h + CurveID	53h	Data Block data (4 bytes)	04h: Data Block is not finished 00h: Data Block is finished D0h: Error: Data Block was already finished

Example
Index = 2001h
Subindex = 53h
Data = 01312D00h

CAN Telegram (8 Byte Data), COB-ID 601, PLC -> LinMot Controller:

	Index		Subindex	Data Block Data			
23	**01**	**20**	**53**	**00**	**2D**	**31**	**01**

Response: (8 Byte Data), COB-ID 581, LinMot Controller -> PLC:

	Index		Subindex	Unused Data		Result	
60	**01**	**20**	**53**	**00**	**00**	**04**	**00**

11. Read curve from controller via CANopen

Get curve

Index	Subindex	Data	Result (4 bytes)
2000h + CurveID	60h	-	00h: Curve exists D4h: Curve does not exist

Example

CuveID = 1

Result = 00 46 1401 -> 00: Curve exists
46: InfoBlock Size bytes
0114: DataBlock Size bytes

Result = D4 xx xxxx -> D4: Curve does not exist

Get Curve Info Block

Index	Subindex	Data	Result (4 bytes)
2000h + CurveID	61h		04h: Info Block is not finished 00h: Info Block is finished D0h: Error: Info Block was already finished

Get Curve Data

Index	Subindex	Data	Result (4 bytes)
2000h + CurveID	62h		04h: Data Block is not finished 00h: Data Block is finished D0h: Error: Data Block was already finished

12. Get UPID List from Controller via CANopen

Start getting UPID List

Index	Subindex	Data	Result
2000h	20h	Start UPID (2 bytes)	00h: OK

Example
Index = 2000h
Subindex = 20h
Start UPID 1000h

CAN Telegram (8 Byte Data), COB-ID 601, PLC -> LinMot Controller:

	Index		Subindex	Unused Data		Start UPID	
23	**00**	**20**	**20**	**00**	**00**	**00**	**10**

Response: (8 Byte Data), COB-ID 581, LinMot Controller -> PLC:

	Index		Subindex	Unused Data		Result	
60	**00**	**20**	**20**	**00**	**00**	**00**	**00**

Get Next UPID List Item

Index	Subindex	Data	Result
2000h	21h	Address Usage	UPID found

Example
Index = 2000h
Subindex = 21h
UPID found = 1004h
Address Usage = 000Dh

CAN Telegram (8 Byte Data), COB-ID 601, PLC -> LinMot Controller:

	Index		Subindex	Unused Data			
40	**00**	**20**	**21**	**00**	**00**	**00**	**00**

Response: (8 Byte Data), COB-ID 581, LinMot Controller -> PLC:

	Index		Subindex	Address Usage		UPID found	
43	**00**	**20**	**21**	**0D**	**00**	**04**	**10**

Address Usage:

			Not used for Hash calculation				Life Parameter					ROM Write	ROM Read	RAM Write	RAM Read
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

The commands for getting the modified UPID List are used the same way.

13. Read the Error Log from the Controller

Get Error Log Entry Counter

Index	Subindex	Data	Result
2000h	70h	-	Number of Logged Errors Number of Occurred Errors

Example
Index = 2000h
Subindex = 70h
Number of Logged Errors = 0015h
Number of Occurred Errors = 0034h

CAN Telegram (8 Byte Data), COB-ID 601, PLC -> LinMot Controller:

	Index		Subindex	Unused Data			
40	**00**	**20**	**70**	**00**	**00**	**00**	**00**

Response: (8 Byte Data), COB-ID 581, LinMot Controller -> PLC:

	Index		Subindex	Nr. Of Logged Errors		Nr. Of Occurred Err	
43	**00**	**20**	**70**	**15**	**00**	**34**	**00**

Get Error Log Entry Error Code

Index	Subindex	Data	Result
2000h + Entry Nr.	71h	-	Error Code

Example
Index = 2005h
Subindex = 71h
Error Code of entry 5 = 64h (Cfg. Err: No Motor defined)

CAN Telegram (8 Byte Data), COB-ID 601, PLC -> LinMot Controller:

	Index		Subindex	Unused Data			
40	**05**	**20**	**71**	**00**	**00**	**00**	**00**

Response: (8 Byte Data), COB-ID 581, LinMot Controller -> PLC:

	Index		Subindex	Unused Data		Error Code	
43	**05**	**20**	**71**	**00**	**00**	**64**	**00**

Get Error Log Entry Time Low

Index	Subindex	Data	Result
2000h + Entry Nr.	72h	-	Time Low (milliseconds)

Example
Index = 2005h
Subindex = 72h
Time Low of entry 5 = 28C1h (=10433ms=10.433s)

CAN Telegram (8 Byte Data), COB-ID 601, PLC -> LinMot Controller:

	Index		Subindex	Unused Data			
40	05	20	72	00	00	00	00

Response: (8 Byte Data), COB-ID 581, LinMot Controller -> PLC:

	Index		Subindex	Time Low		Time Mid Low	
43	05	20	72	C1	28	00	00

Get Error Log Entry Time High

Index	Subindex	Data	Result
2000h + Entry Nr.	73h	-	Time High (hours)

Example
Index = 2005h
Subindex = 73h
Time High of entry 5 = 0398h (=920 hours)

CAN Telegram (8 Byte Data), COB-ID 601, PLC -> LinMot Controller:

	Index		Subindex	Unused Data			
40	05	20	73	00	00	00	00

Response: (8 Byte Data), COB-ID 581, LinMot Controller -> PLC:

	Index		Subindex	Time Mid High		Time High	
43	05	20	73	98	03	00	00

The Time of an entry consists of 32Bit hours and 32Bit milliseconds.

Get Error Code Text Stringlet

Index	Subindex	Data	Result
2000h + Error Code.	74h + (Stringlet Nr. 0..7)	-	4 Bytes of Error Code Text

Example
Index = 2064h (Error Code 64h = „Cfg Err: No Motor Defined“)
Subindex = 74h
Character 0..3 = 43 66 67 20 = “Cfg “

CAN Telegram (8 Byte Data), COB-ID 601, PLC -> LinMot Controller:

	Index		Subindex	Unused Data			
40	64	20	74	00	00	00	00

Response: (8 Byte Data), COB-ID 581, LinMot Controller -> PLC:

	Index		Subindex	Char 0	Char 1	Char 2	Char 3
43	64	20	74	43	66	67	20

The Time of an entry consists of 32Bit hours and 32Bit milliseconds.

14. Reset Parameters to default values

There are three options to reset the parameters of a LinMot E1100 controller to default values:

1.) By manipulating the two rotary hex switches:
- Power Off the controller
- Set the switches to FF
- Power On the controller
- Set the switches to 00
- Wait for 10 s
- Power Off the controller

2.) By writing Index 0x2000h subindex 7h, 8h, 9h, Ah of the Objectdictionary.
After changing the ROM values, a Reset should be performed either by a NMT Reset command or by Power OFF and ON the controller.

3.) Reinstall the firmware will always reset the parameters to default values

15. Example

The following example shows the homing procedure and execution of a motion command by CANopen:

The PDO mapping is default:
Receive PDO's:

R_PDO1	**R_PDO2**	**R_PDO3**
Control Word	CMD Header	CMD Slave Header
	Par 1	Par 4
	Par 2	Direct Par Channel 1
	Par 3	Direct Par Channel 2

Homing (Control Word = 0x083Fh)

R_PDO1	
Byte Nr.	Value
0	3Fh
1	08h
2	X
3	X
4	X
5	X
6	X
7	X

Enter Operational State (Control Word = 0x003Fh)

R_PDO1	
Byte Nr.	Value
0	3Fh
1	00h
2	X
3	X
4	X
5	X
6	X
7	X

Execute Motion Command : VAI 16Bit Go To Pos (090xh)

Target Position :	50mm		01F4h
Maximal Velocity :	1m/s		03E8h
Acceleration :		10m/s^2	0064h
Deceleration :		10m/s^2	0064h

R_PDO2	
Byte Nr.	Value
0	01h
1	09h
2	F4h
3	01h
4	E8h
5	03h
6	64h
7	00h

R_PDO3	
Byte Nr.	Value
0	01h
1	09h
2	64h
3	00h
4	X
5	X
6	X
7	X

As it appears with LinMot-Talk1100 after «Read Command» in the Control Panel :