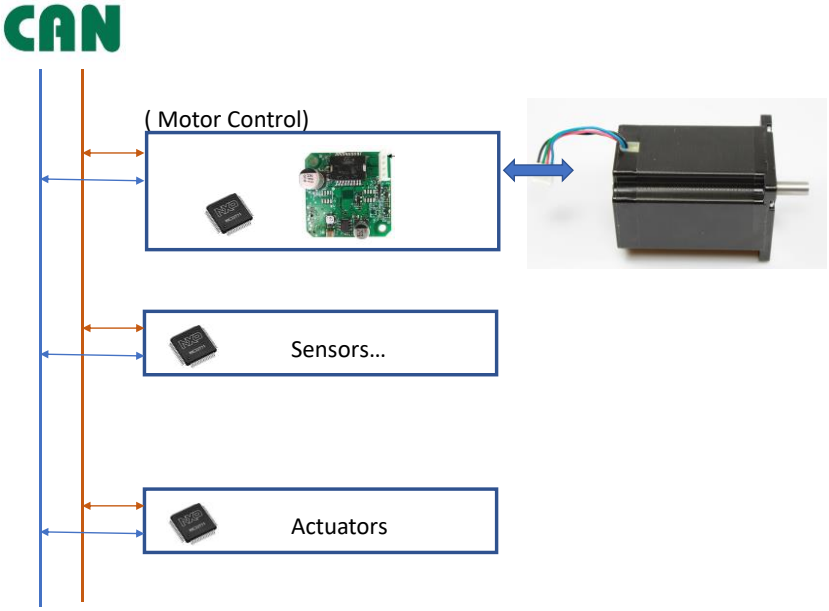
KANN–CANopen protocol

V2.3 – 7.06.2022

M. Zimmermann

This document is part of the software documentation. It explains special aspects of the CAN-CANopen topic including how to update by a bootloader.

System overview



Contents

1. History

Version	Datum	Description
A	14.12.17	1. Version of the functional model
B	11.4.2018	2. Extensions regarding reference drive & default values
C	16.4.2018	3. Restructured and supplemented, PDO's set for an efficient and simple control, see chapter 5-6
D	25.4.2018	4. small corrections, SDO 0x2020 supplemented, SDO 2011-6 supplemented, error codes supplemented
E	25.5.2018	**From Firmware V0.3-0** the following items are integrated: Object 2010, extended with Control-Mode and Holding Torque Switching of 0.1° Mode, moreover holding torque now possible Object 2013, CMD for actual position implemented as homing mark
V1.1	15.6.2018	Ranges with the Parameter defined
V1.2	2.4.2019	From Firmware V0.9-007, changes of the parameters of the KannMotion line - Object 1018, modified on Standard Info KannMotion - Object 2010, modified on Standard Params KannMotion - Object 2013 changed - Diagnosis Object 2030 adapted - Objects 2014/2015 new implemented
V1.3	18.10.2019 29.10.2019	From Firmware V0.9-011, additional parameters possible - Object 2020, extended with onTimerEvent and ComWatchdogTime Confusing green color of DSP402 removed
V1.4	4.3.2020	English version
V2.0	6.11.2020 New Features coming with FW V2.0-002	***KannMotion gets closer with CANopen CiA301 / CiA402*** ***New:*** - SDO 0x1010 New, added - SDO 0x1400-1403 New, added - SDO 0x1600-1603 New, added - SDO 0x1800-1803 New, added - SDO 0x2014 'Save Marked Position as Home' added - SDO 0x2018-2019 New, added - SDO 0x603F New, added - SDO 0x6040 New, added - SDO 0x6041 New, added - SDO 0x6044 New, added - SDO 0x6081 New, added - NMT-Management chapter added ***Changes:*** - SDO 0x2015 changed to USER c-code Info - SDO 0x2020 changed, less parameters - SDO 0x2020:3 .. Control bit for NMT Startup Behave (PreOperatinal or Operational) - SDO error messages: error codes changed - PDO- Mapping - PDO-SYNC integrated

V2.1	23.2.2021 11.8.2021	SDO2014 Information extended SDO6067/6068 Defaults in Table corrected
V2.2	17.11.2021 08.03.2022 22.03.2022	SDO2017 extended SDO2033 Extended specification of 2010-11 and 2011-6 Sample of PDO activation included
V2.3	7.06.2022	SDO 1008, 1009, 100A added

2. Introduction in the CAN-CANopen communication

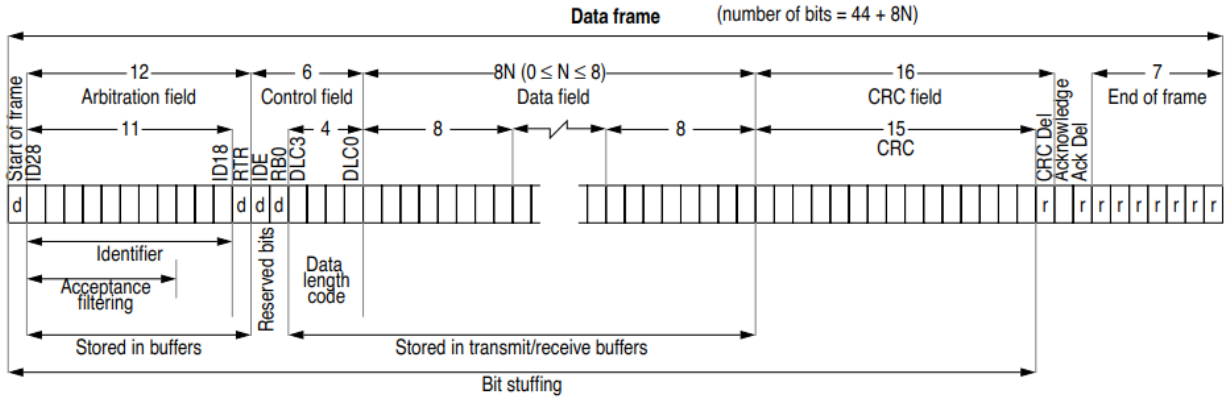
Bit rate

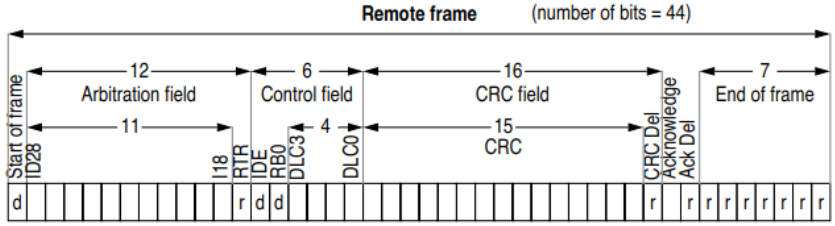
The bit rate can be adjusted to **125k /250k/500k**/750k/1M.

Table: usual bit rates CANopen:

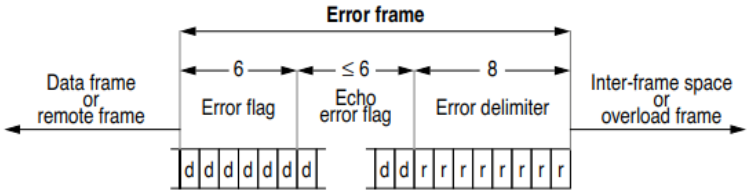
Bit Rate	Max Bus Length (m)	Max Drop Length (m)	Max Cumulative Drop Length (m)
1M	25*	2	10
800k	50*	3	15
500k	100	6	30
250k	250	12	60
125k	500	24	120
50k	1000	60	300
20k	2500	150	750
10k	5000	300	1500

CAN Frame types

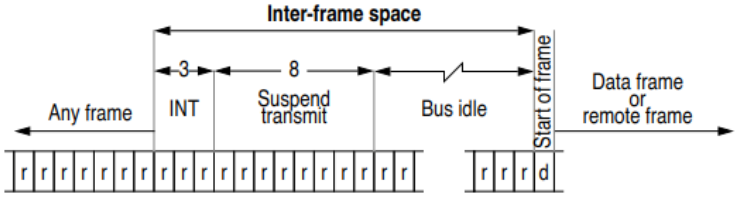




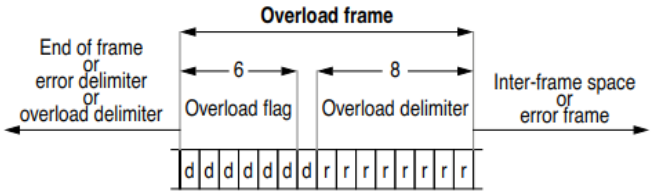
Note: A remote frame is identical to a data frame, except that the RTR bit is recessive, and there is no data field.



Note: An error frame can start anywhere in the middle of a frame.

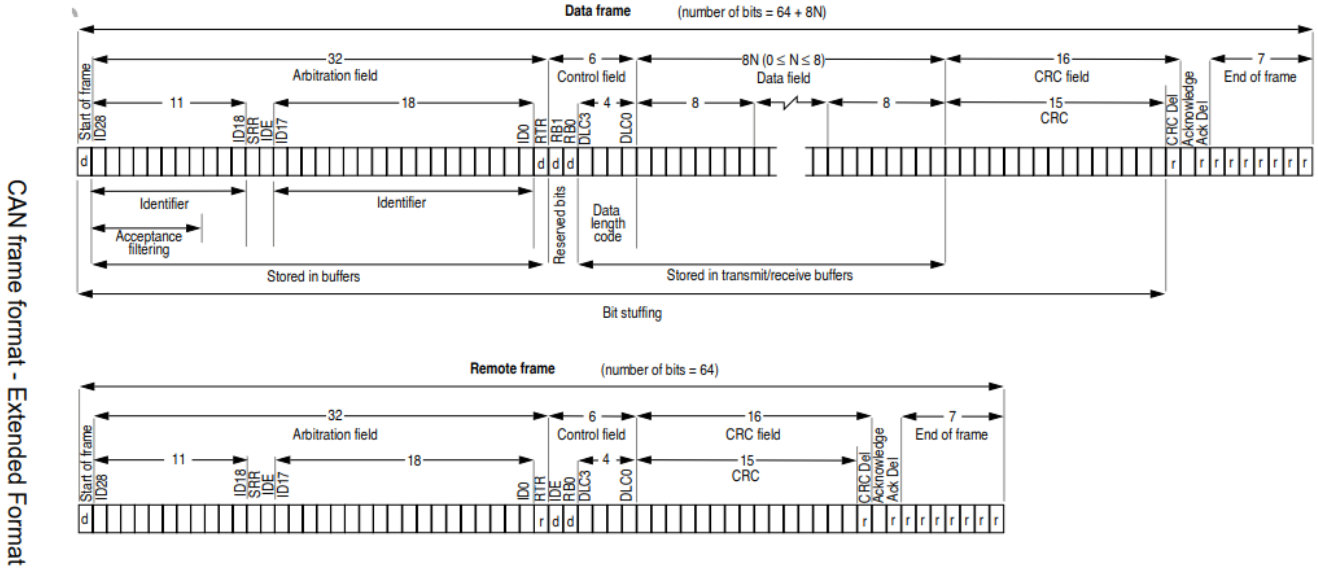


Note: INT = Intermission
Suspend transmission is only for error passive nodes.



Note: An overload frame can only start at the end of a frame.
Maximum echo of overload flag is one bit.

Extended Format



Note: A remote frame is identical to a data frame, except that the RTR bit is recessive, and there is no data field.

Applied Frame types

data frames

Addressing (ID's)

CANopen Default Identifier Distribution

A default indentifier distribuciton saves configuration expenses. Thereby the node number is embedded in the identifier. The default indentifier distribuciton is defined as follows:

Identifier 11-Bit (binär)	Identifier (dezimal)	Identifier (hexadezimal)	Funktion
00000000000	0	0	Netzwerkmanagement
00010000000	128	80h	Synchronisation
0001xxxxxxx	129 - 255	81h - FFh	Emergency
0011xxxxxxx	385 - 511	181h - 1FFh	PDO1 (tx)
0100xxxxxxx	513 - 639	201h - 27Fh	PDO1 (rx)
0101xxxxxxx	641 - 767	281h - 2FFh	PDO2 (tx)
0110xxxxxxx	769 - 895	301h - 37Fh	PDO2 (rx)
0111xxxxxxx	897 - 1023	381h - 3FFh	PDO3 (tx)
1000xxxxxxx	1025 - 1151	401h - 47Fh	PDO3 (rx)
1001xxxxxxx	1153 - 1279	481h - 4FFh	PDO4 (tx)
1010xxxxxxx	1281 - 1407	501h - 57Fh	PDO4 (rx)
1011xxxxxxx	1409 - 1535	581h - 5FFh	SDO senden
1100xxxxxxx	1537 - 1663	601h - 67Fh	SDO empfangen
1110xxxxxxx	1793 - 1919	701h - 77Fh	NMT Error Controll
xxxxxxx = Knotennummer 1 - 127			

Tabelle 2.5: Default-Identifier

CANopen allows a completely free identifier configuration.

Within the pre-defined connection set, the following identifiers are not used:

COB-ID (Identifier values)
001_h - $07F_h$
101_h - 180_h
200_h / 280_h / 300_h / 380_h / 400_h / 480_h / 500_h / 580_h
600_h / 680_h
780_h - $7E3_h$
$7E6_h$ - $7FF_h$

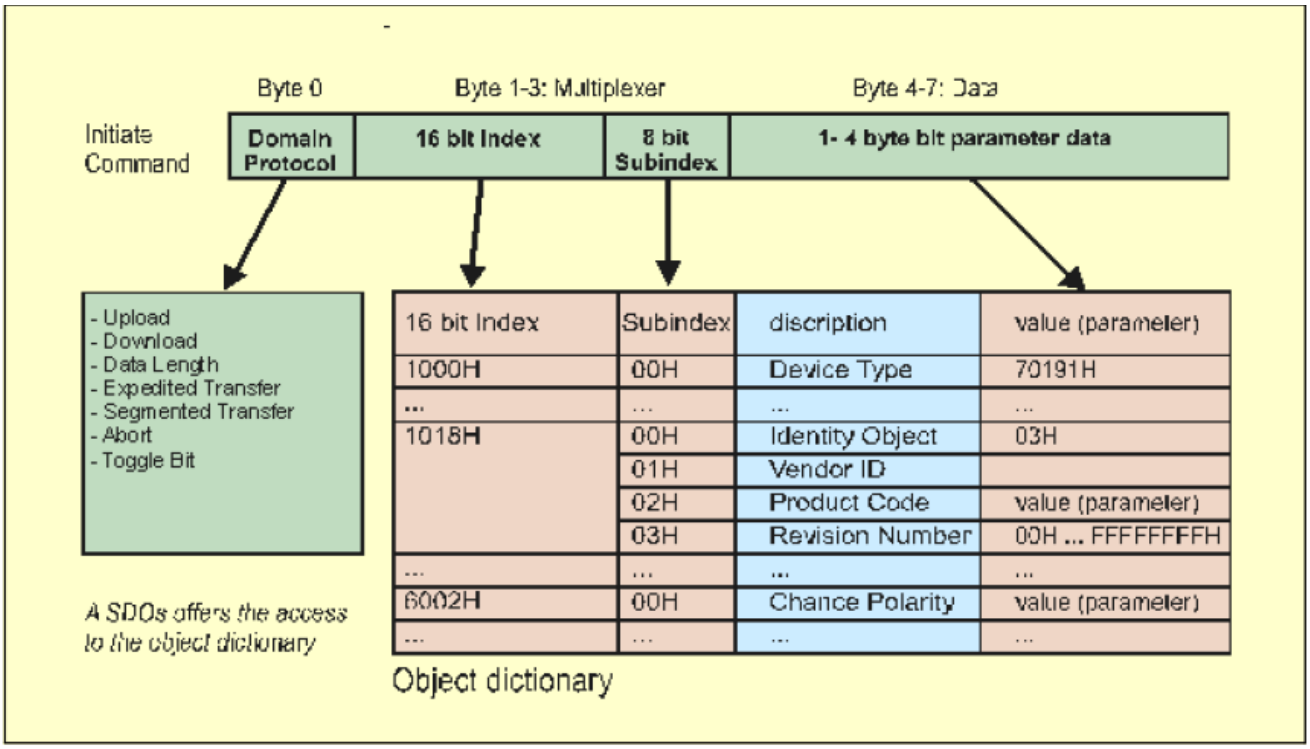
Table 2: Unused identifiers in the Pre-defined Connection Set

Byte ordering: In multibyte variables, the bytes are ordered by significance – lowest significant byte comes first.

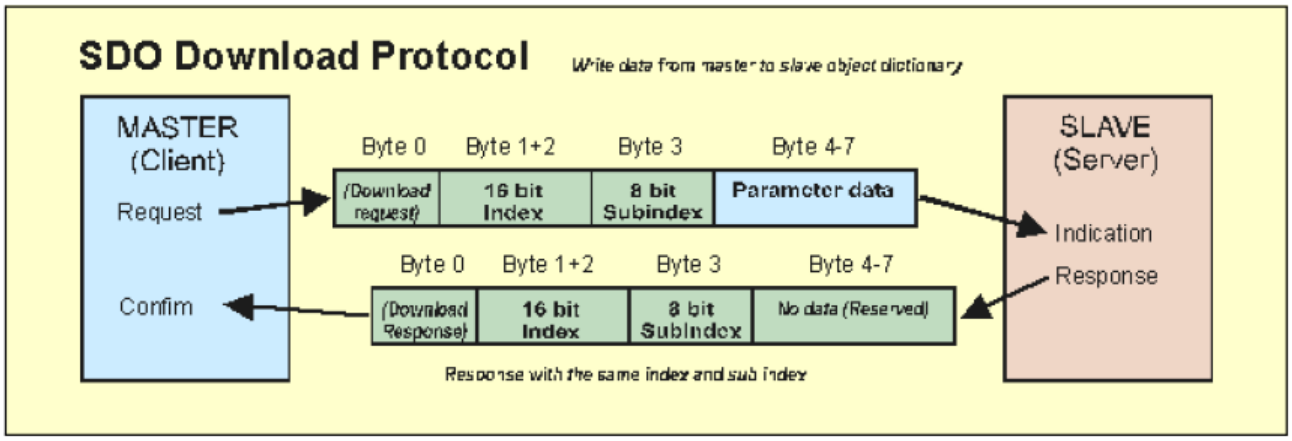
CANopen objects

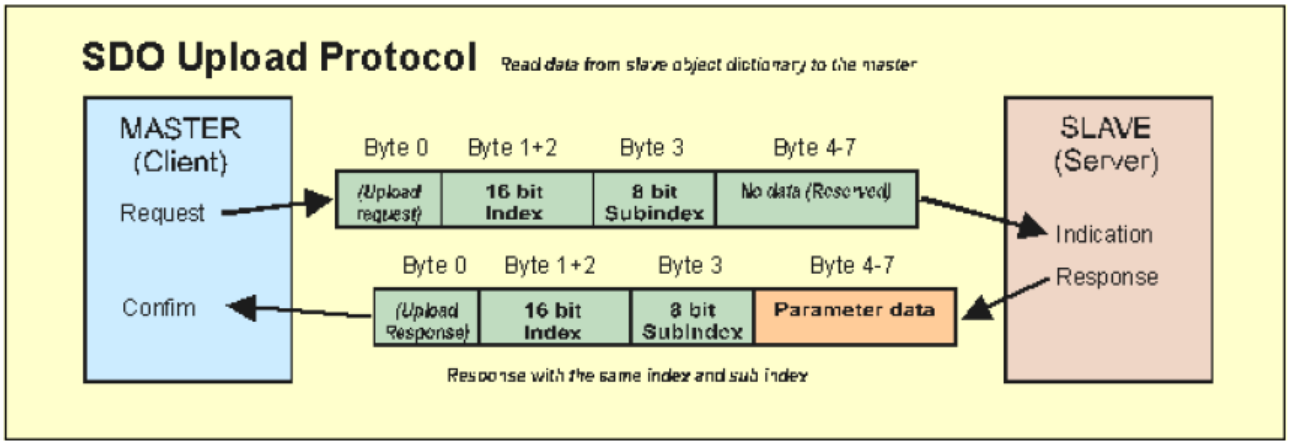
SDO objects (service data object)

A service data object contains a „Domain protocol (8-Bit)“, the „Index (16-Bit)“, the „Sub-Index (8-Bit)“, and up to 4 databytes. The domain protocol contains the action, that follows the parameter, to which refer the index and the sub-index. Are the parameters supposed to get new values, these values can be transmitted in the data bytes.



The 8 bytes of the SDO (as illustrated here) are hosted in the data area of the CAN-Message. The adressing of the node conists of two telegrams at least.





Command	Access to Data 1 - Data 4			Block
	4 byte data (5th - 8th byte)	2 byte data (5th + 6th byte)	1 byte data (5th byte)	
	hex	hex	hex	hex
Write request (Send parameters to drive)	23	2B	2F	Writing not possible
Write Response (Controller response to the write request (acknowledgement))	60	60	60	
Read Request (Request to read a parameter from the drive)	40	40	40	40
Read Response (Response to the read request with an actual value)	43	4B	4F	41
Error response (The controller indicates a communication error)	80	80	80	80

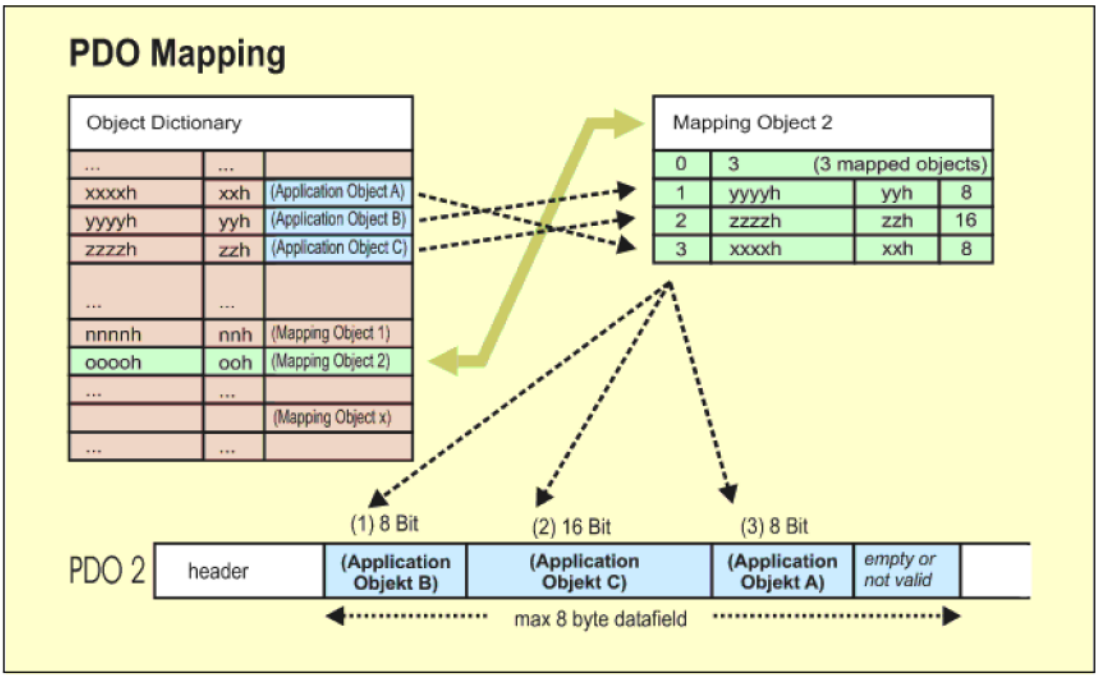
General assignment of the object-index-numbers

Object Index (hex)	Object
0000	Not used
0001 - 001F	Static Data Types
0020 - 003F	Complex Data Types
0040 - 005F	Manufacturer Specific Complex Data Types
0060 - 007F	Device Profile Specific Static Data Types
0080 - 009F	Device Profile Specific Complex Data Types
00A0 - 0FFF	Reserved for further use
1000 - 1FFF	Communication Profile Area
2000 - 5FFF	Manufacturer Specific Profile Area
6000 - 9FFF	Standardized Device Profile Area
A000 - FFFF	Reserved for further use

PDO objects (prozess data objects)

The exchange of process data with CANopen is effected via the CAN-bus, meaning without protocol-overhead. The broadcast-characteristic of the CAN-bus remains completely. A message therefore can be received by all nodes and can be evaluated (producer-consumer-model).
Because the protocol structure in the telegram is missing, the participant of the bus (for whom the data is intended) must know how the information is embedded in the data area of the PDO (which bit/byte is which value). This declaration therefore is affected in advance during the initialization of the net by the so-called PDO-mapping. This allows to place the required information at a certain point in the data area of a PDO.

In order to enable a variable configuration of the PDO data, the mapping itself is effected at a special mapping object. This mapping is in principle a table, in which the objects, that shall be mapped, can be entered.



3. implementation of CANopen

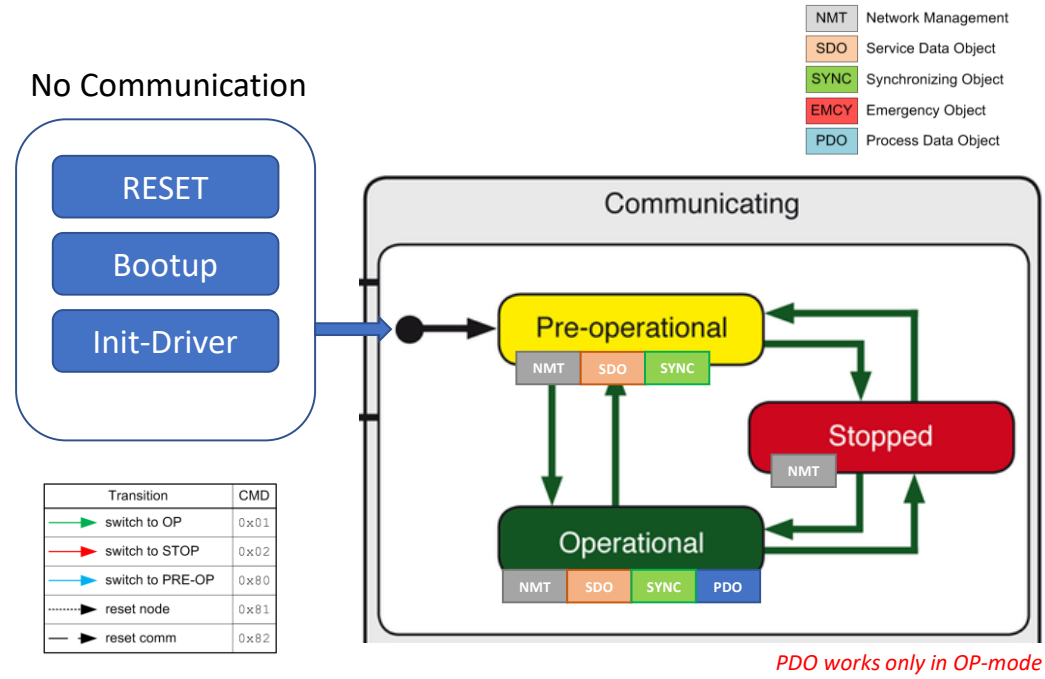
The KannMOTION comes with included of CiA301 and CiA402 Standard, extended by some proprietary objects. PDO mapping is shown by SDO, and is fix mapped. MPDOs and dynamic SDO-PDO mapping is dispended.
As a Highlight, PDO4 are forwarded to the User specific code section of KannMOTION, so this enables lot of customized wishes to be realized.

Factory Default

A new KannMOTION Drive will have following Preset:

Bitrate: ***250k***
Node Address: ***3***
Heartbeat: ***2500ms / ON***

Network Management



PDO works only in OP-mode

Network Message, Data length 2-Bytes

CAN ID	Data	
	Byte 0	Byte 1
000	<CMD>	<NodeID>

Byte1: <NodeID> 0x00 -> Broadcast
else -> Message only for device with NodeID

Byte0: <CMD>

CMD	Description
0x01	Switch to Operational
0x02	Switch to Stop
0x80	Switch to Pre-Operational
0x81	Reset-Node (Restart Device)
0x82	Reset Communication (Restart of CAN Module)

NMT State is reportet by Heartbeat

Heartbeat producer

The Heartbeat is showing the actual NMT State

Heartbeat Value	Description
0	Boot-up
1	Off bus
4	Stopped
5	Operational
0x7F	Pre-operational

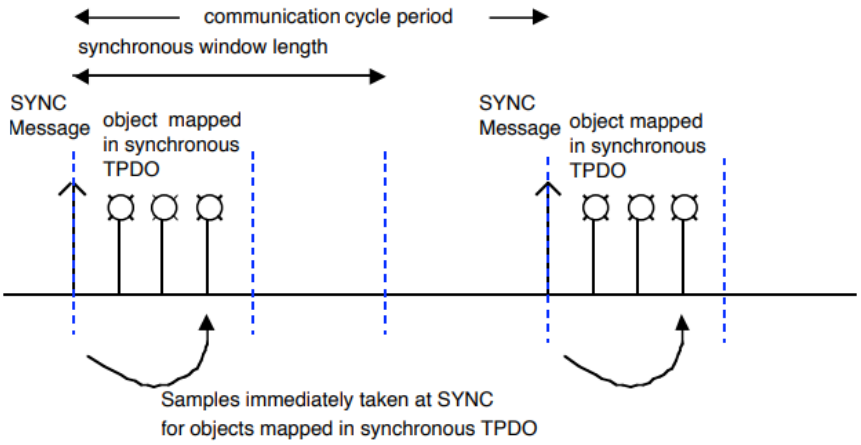
Example

Msg.-Type	Id	DLC	Data (Hex)	Data (ASCII)
STD	703	1	00	.
STD	703	1	05	.
STD	703	1	05	.
STD	703	1	05	.

Sync protocol

CAN-ID	Data		Description
0x80	No data	*(DLC=0)*	Standard sync, see also SDO 1800..1803
0x80	SYNC counter	*(DLC=1)*	Sync Groups

Sync producer might send SYNC message with DLC=0 or DLC=1, in case of DLC=1, only objects are synchronized where SYNC_counter = Content of 0x1800:06.



4. Implemented SDO's (Service Data Object)

SDO-accesses (read and write) to the object list are executed by SDO-telegram. The 8 data bytes are split in 4 bytes for the addressing and 4 bytes for user data. SDO-accesses are always responded. At data sizes > 1-Byte <Liddle Endian> is applied in accordance with its specifications. That is to say, the lowest byte is transmitted first.

SDO-object-list

Index (hex)	Sub-Index	Name	Size (bit)	Default Value	Access	Non volatile	Description	PDO-Mapping
1000	0	**Device Type**	u32		ro		Device Type / Stepper 0x00040192	
1001	0	**Error Register**	u8	0	ro		**Bit Meaning** 0: generic error (app not in RUN) 1: current 2: voltage 3: temperature 4: communication error 5: device profile specific 6: Reserved (always 0) 7: Encoder Error	TPDO-2
1005		**COB-ID SYNC message**	u32	0x80	ro		SYNC COB-ID, only consumer	
1008 [5]		**manufacturer device name**	vstr		ro		Device Name	
1009 [5]		**manufacturer hardware version**	vstr		ro		Hardware (as Text)	
100A [5]		**manufacturer software version**	vstr		ro		Firmware (as Text)	
1010	0	**Store parameters** No. of subentries	u8	4	ro			
	1	Save all parameters	u32	0x0001	rw		Write Data = 'Save' = 0x65766173	
	2	Save communication parameters	u32	0x0001	rw			
	3	Save application parameters	u32	0x0001	rw			
	4	Save KannMotion Drive Params	u32	0x0001	rw			
1011	0	Restore default parameters No. of subentries	u8	3	ro			
1017	0	**Heartbeat Time**	u16	2500	rw	YES[4]	**Heartbeat Time in ms** 0: off 20…15000	
1018	0	**Identify Object** / No. of subentries	u8	**6**	ro			
	1	Vendor ID	u32		ro	YES		
	2	Hardware ID Number	u32		ro	YES	ArtNbr: PCB-Board / including Index	
	3	System ID-Number	u32		ro	YES	ArtNbr: Motor complete	
	4	Serial number	u32		ro	YES		
	5	Firmware Revision	u16		ro	YES		
	6	Firmware ID Number	u32		ro	YES		
1400-1403		**Receive PDO Communication Parameter**		**2**	ro	YES[4]	See CiA 301 or *SDO 0x1400…0x1403, Receive PDO-Communication* Parameter	
1600-1603		**Receive PDO Mapping Parameter**		0..3	ro		We use Fixed predefined mapping See CiA 301 or *Activation Example (Step by Step)*	
1800-1803		**Transmit PDO Communication Parameter**		**5**	ro	YES[4]	See CiA 301 or *SDO 0x1800…0x1803, Transmit PDO- Communication* Parameter	
1A00-1A03		**Transmit PDO Mapping Parameter**		0..3	ro		We use Fixed predefined mapping See CiA 301 or *Activation Example (Step by Step)*	

.[4] value might be packed, means there might be a loss in density especially at 16-Bit values, after restart/read!
. [5] available depending on firmware revision

Index (hex)	Sub-Index	Name	Size (bit)	Default Value	Access	Non volatile	Description	PDO-Mapping
2010	0	**Motor Drive setup Object** No. of subentries	u8	19	ro		Motor settings 'Driver-Side'	
	1	Driver Factory Setting	u8	0	ro	YES	Factory Settings	
	2	MotStepType	u8	0	ro	YES	12:[15°];24:[7.5°];100:[1.8°];200:[0.9°]	
	3	OutConfig[1)]	u8	0	ro			
	4	Driving Current max.	u16		ro	YES	Default current 100% in [mA] [0..2700]	
	5	Acceleration max	u16		ro	YES	Maximum acceleration [r/s2] [0..0xFF.FF] FixPoint8	
	6	Din Low Threshold[1)]	u16	5000	rw	YES	Input Threshold for digital Inputs [mV]	
	7	Din High Threshold[1)]	u16	15000	rw	YES	Input Threshold for digital Inputs [mV]	
	8	Gear Ratio	u32	1.0 [16777216]	rw	YES	Gear Ratio [0..0xFF.FFFFFF] FixPoint24 in **[mm/Round]** or in **[AxisRound/MotorRound]**	
	9	Gear backlash	s16	0	rw	YES	Back lash from gear box [+/- 32767] in Microsteps	
	10	Control Mode	u8	0	rw	YES	Control Mode [0..1] [0]: Control in micrometer [um] [1¡: Control in 1/10° [0.1°]	
	11	Micro steps count	u8	4	rw	YES	Value Range: 0..7 MicroSteps = 2^{Value} -> **4** = 2^4 = **16** Micro steps Available Microsteps: [1,2,4,8,16,32,64,128]	
	12	Min. speed 0.1 rpm	u16	250	rw	YES	[0.1rpm] e.g. 102 = 10.2 rpm	
	13	Max. speed 0.1 rpm	u16	1500	rw	YES	[0.1rpm]	
	14	Torque-HOLD	u8	0	rw	YES	[0.5%], Holding Torque	
	15	Torque-ACC[2)]	u8	200	rw	YES	[0.5%], ACC Torque	
	16	Torque-RUN	u8	200	rw	YES	[0.5%], RUN Torque	
	17	Torque-DEC[2)]	u8	200	rw	YES	[0.5%], DEC Torque	
	18	Acceleration	u8	200	rw	YES	[0.5%], Acceleration	
	19	Deceleration	u8	100	rw	YES	[0.5%], Deceleration	
	20	Position Regulator Control	u8	0	rw	YES	Position Regulator Control Byte	
	21	Driver Input Filter Control	u8	0	rw	YES	Analog and Digital Input Filter Cntrl	
2011	0	**Motor Control Object** / No. of subentries	u8	**9**	ro		Motor Driver State 'Driver-Side'	
	1	Motor State	u8		ro		**eMS_IDLE = 0,** //!< Motor-Driver is in Standby / Sleep / .. now torque **eMS_HOLD = 1,** //!< Motor-Driver is on, Motor is in stand still **eMS_ROTATE = 0x10,** //!< Start rotation **eMS_GOTOPOS = 0x20,** //!< drive to Position **eMS_DoSTOP = 0x80,** //!< Stop	TPDO-2
	2	Target position	s32	0	rw		Micro-Steps	
	3	Actual position	s32	0	ro		Micro-Steps	
	4	Target speed	s16	0	rw		0.1-rpm	RPDO-2
	5	Encoder position	s32	0	ro		Micro-Steps	
	6	Regulator Bits	u32	0	ro		Info-Bits B0: Closed Loop active B8: Encoder Error B9: Pos Error B10: Homing RUN active B11: Homing shall be Executed (Suggestion) B12: Gear-Calculation Error B13: NewStartNeeded B15: Target Position reached B16: HomingEnd B17: Target Position changed B20: Stall-detected	

	7	Main State	u8		ro			
	8	Temperature	u8		ro		Temp[°C] = (Return Value – 50)	TPDO-2
	9	Errorbits	u8		ro		Lower-8-Bits of 0x603F	
2012	0	Not used	u8		ro			-
2013	0	**Command AND Setting Object** / CMD	u8	**0**	rw		0x00: no CMD 0x01: Goto Home 0x02: Fast-Stop 0x03: Soft-Stop/Rotate 0x04: Store all NV Params into NV .. 0x0F: Clear Error / Retry 0x10: Activate Closed-Loop 0x11: Disable Closed-Loop *While Read: Bit4 shows Closed loop* .. 0x40: Set Actual Pos as Home *see also 0x2014:01*	RPDO-2
2014	0	**Homing Control**	u8	3	ro			
	1	Homing Mode	u8	0	rw		Mode see Homing description Set Sub 2 & 3 before Sub 1 will Start Homing Movement	
	2	Homing Timeout [ms]	u16	10000	rw		Timeout in [ms] *0xFFFF = no Timeout*[5)]	
	3	Homing Speed [rpm]	S8	100	rw		Speed and Direction 1rpm/E `[-:CW] /[+:CCW]`	
2015	0	**ADLOS User C-Info**	u8	**8**	ro		ADLOS User C-Info	
	1	UserPrgVersion	u32	0xFFFF	ro		Users specific 'C-Program' Version	
	2	u8_UserPrgTxt Char[1..4]	u32	0xFFFFFFFF	ro		User 'C-Program' Text char [1..4]	
	3	u8_UserPrgTxt Char [5..8]	u32	0xFFFFFFFF	ro		User 'C-Program' Text char [5..8]	
	4..8	u8_UserPrgTxt Char[9..28]	u32	0xFFFFFFFF	ro		User 'C-Program' Text char [9..28]	
2016	0	**ADLOS User VAR, Debug**	u8	**11/20**	ro		ADLOS Access USER RAM	
	1	u16_Timer5ms[0]	u16	0	ro		User Timer	
	2	u16_Timer5ms[1]	u16	0	ro		User Timer	
	3	u16_Timer5ms[2]	u16	0	ro		User Timer	
	4	u16_Timer5ms[3]	u16	0	ro		User Timer	
	5	u8_StepChain[0]	u8	0	ro		User StepChain	
	6	u8_StepChain[1]	u8	0	ro		User StepChain	
	7	u8_StepChain[2]	u8	0	ro		User StepChain	
	8	u8_StepChain[3]	u8	0	ro		User StepChain	
	9	i32_PosVAR_um_01deg[0]	s32	0	ro		User Position Var	
	10	i32_PosVAR_um_01deg[1]	s32	0	ro		User Position Var	
	11	i32_PosVAR_um_01deg[2]	s32	0	ro		User Position Var	
2017	0	**Maintainance**	UI_8	**7 / 10**[5)]			ADLOS Maintainance Data	
	1	Total Runtime	u32	0	r	YES *0x0A0*	in [s] 1=1 second	
	2	Critical Temperature Time	u16	0	r	YES	in [s] 1=1 second	
	3	OverTempErrCnt	u16		r	YES	[Events]	
	4	OverCurrErrCnt	u16		r	YES	[Events]	
	5	TimeOutErrCnt	u16		r	YES	[Events]	
5)	6	Stall Error Counter	u16		r	YES	[Events]	
5)	7	U-motor Error Counter	u16		r	YES	[Events]	
5)	8	Driver Chip Error Counter	u16		r	YES	[Events]	
	6 / 9	MinTemperature	UI_8		r	YES	In [°C] w. Offset of 50°C	
	7 / 10	MaxTemperature	UI_8		r	YES	In [°C] w. Offset of 50°C	
2018	0	**DEBUG USER NV-Data**		16			ADLOS Access USER NV	
2019	0	**DEBUG USER DATA**		128			ADLOS Access USER RAM	
2020	0	**CAN Control Object**	u8	4	ro			
	1	Bitrate	u8	1	rw	YES	0:125k/1:250k/2:500k	
	2	NodeAdress	u8	3	rw	YES	1..127	
	3	Control bits	u8	0	rw	YES	B0: NMT-changes automatic into OP-Mode after Reset B4: Closed Loop Position-Control	
	4	ComWatchDogTime [ms]	U16	0	rw	YES[4]	0: off / 50…65535 ms	
2030	0	**Diagnostic EncoderData** Number of subentries	u8	**3**	ro		Encoder Info *.. 12Bit masked*	

	1	Angle	u16		ro		Angle abs	
	2	State	u16		ro		State-Bits	
	3	Temperature	u16		ro		Temperature	
	4	FieldStrength	u16		ro		FieldStrength in Gaus	
	5	Turns	u16		ro		TurnsCounter	
2033	0	**Controller Specific Online Data** Number of subentries	u8	**2..**	ro		Online Data *.. Like Input States* *.. ADC Values* *see specific Controller description*	
	1	Input States	U8		ro		Inputs States-Bit Field	
	2	ADC_value-X	u16		ro		State-Bits	

.[1)] onyl effective if IO in hardware exists
.[2)] depending on hardware / firmware not used
. [5)] available depending on firmware revision

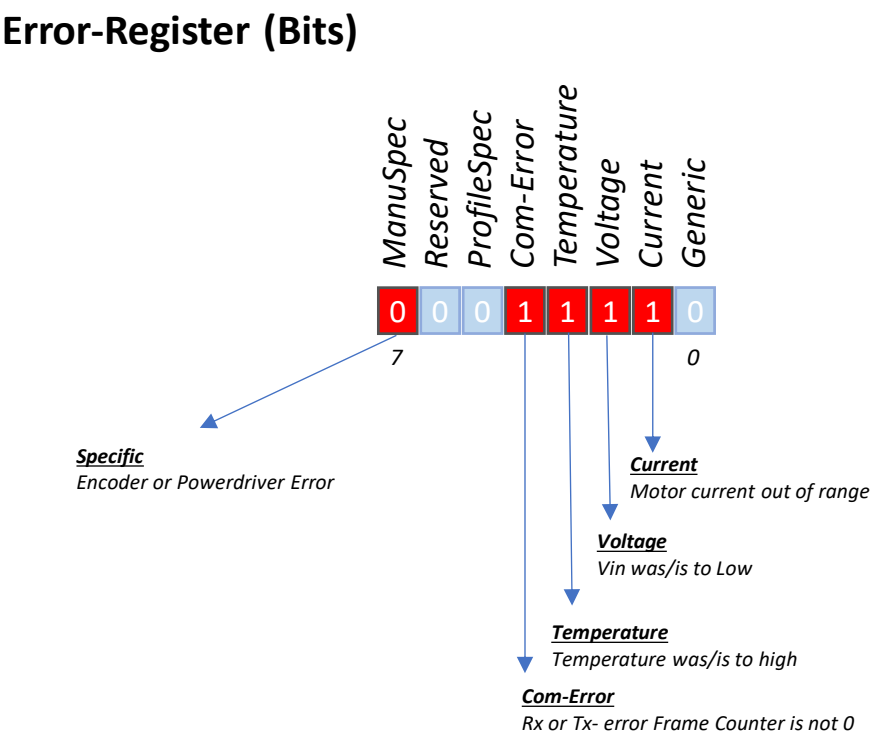
Index (hex)	Sub-Index	Name	Size (bit)	Default Value	Access	Non volatile	Description	PDO-Mapping
603F	0	**ErrorCode**	u16		ro	X	Error Code	TPDO-2
6040	0	**Controlword**	u16		rw	?	Drive Control-Bits	
6041	0	**Statusword**	u16		ro		Drive State-Bits	TPDO-1
6044	0	**Velocity control effort (actual value)**	i16		ro		Velocity actual value [0.1rpm]	TPDO-1
6060	0	**Modes of operation**	u8		rw		Set operation Mode: 1: Profile Position Mode PP 3: Profile Velocity Mode 6: Homing Mode	
6061	0	**Modes of operation display**	u8		ro		Read Active operation Mode	
6064	0	**actual Position**	i32	0	ro		Actual position in **[1um]** or **[0.1 °]** *depending on 6041-Bit8*	TPDO-1
6067		**Position Window**	U32	9	rw	YES	Target position tolerance in **[1um]** or **[0.1 °]** *depending on 6041-Bit8* 2…4000	
6068		**Position Window Time**	U16	50	rw	YES	Target position window time in **[ms]** [2] 10…2000	
607A	0	**Target Position**	i32	0	rw		Target position in **[1um]** or **[0.1 °]** *depending on 6041-Bit8*	RPDO-1
607D	0	**Position Limit** No. of subentries	u8	**2**	ro		Limits Object	
	1	Min Position Limit	i32	0	rw	YES	Min Limit in **[1um]** or **[0.1 °]** *depending on 6041-Bit8*	
	2	Max Position Limit	i32	30000	rw	YES	Max Limit in **[1um]** or **[0.1 °]** *depending on 6041-Bit8*	
607F	0	**Max Profile Velocity**	U32		rw	YES [3)]	The max profile velocity is the maximum allowed speed coupled with 2010:13 in **[0.1rpm/E]**	
6081	0	**Profile Velocity**	U32		rw		velocity normally attained at the end of the acceleration ramp during a profiled move and is valid for both directions of motion in **[0.1rpm/E]**	RPDO-3

6083	0	**Profile Acceleration**	U32		rw	YES [3]	Acceleration *same as [0x2010:18]* in **[0.5%]** of [0x2010:5]	
6084	0	**Profile Deceleration**	U32		rw	YES [3]	Deceleration *same as [0x2010:19]* in **[0.5%]** of [0x2010:5]	

.[1] only effective if IO in hardware exists
.[2] depending on hardware / firmware not used
.[3] store instruction needed
.[4] value might be packed, means there might be a loss in density especially at 16-Bit values, after restart/read !

SDO-Details

SDO 0x1001 Error Register



SDO 0x1017 Producer Heartbeat Time

The heartbeat is transferred to the according identifier. The slaves report periodically to the master (the period is adjustable, which is helpful especially for debugging). In this way the availability of the system is checked. The heartbeat time can be adjusted or turned off with SDO 0x1017.

Slave Heartbeat

The slave heartbeat has the following characteristics:

- Produced by slave nodes on the network
- Consumed by the CANopen master
- The COB ID range is in the range 0x701 – 0x77F
- The data frame is 1 byte in length and contains a description of the slave node's communication state according to the table below:

Heartbeat Value	Description
0	Boot-up
1	Off bus
4	Stopped
5	Operational
0x7F	Pre-operational

COB ID	DLC	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
0x701	1	05	-	-	-	-	-	-	-

SDO 0x1400…0x1403, Receive PDO- Communication Parameter

Defines the functionality of a PDO. Due to the fact of fixed PDO Mapping these SDOs are Read only.

Object-ID				Sub-Index			Description
0x1400	0x1401	0x1402	0x1403				
2	2	2	2	**0**	*U8*	*ro*	Largest sub-index supported
0x4000'0200 +Node	*0x4000'0300 +Node*	*0x4000'0400 +Node*	*0x4000'0500 +Node*	**1**	*U32*	*ro*	PDO COB-ID
0xFE	*0xFE*	*0xFE*	*0xFE*	**2**	*U8*	*rw*	Transmission type *[0 or 0xFE]*

Structure of PDO COB-ID entry

UNSIGNED32

	MSB				LSB
bits	*31*	*30*	*29*	*28-11*	*10-0*
11-bit-ID	0/1	0/1	0	00000000000000000	11-bit Identifier
29-bit-ID	0/1	0/1	1	29-bit Identifier	

PDO COB-ID entry, Bit representation

bit number	value	meaning
31 (MSB)	0	PDO exists / is valid
	1	PDO does not exist / is not valid
30	0	RTR allowed on this PDO
	1	no RTR allowed on this PDO
29	0	11-bit ID (CAN 2.0A)
	1	29-bit ID (CAN 2.0B)
28 – 11	0	if bit 29=0
	X	if bit 29=1: bits 28-11 of 29-bit-COB-ID
10-0 (LSB)	X	bits 10-0 of COB-ID

PDO Transmission types

transmission type	PDO transmission				
	cyclic	acyclic	synchronous	asynchronous	RTR only
0		X	X		
1-240	X		X		
241-251	- reserved -				
252			X		X
253				X	X
254				X	
255				X	

Transmission type	Description
0	Data will be copied into PDO while receiving a Sync, PDO is sent/executed afterwards Event occurs
1-240	Data will be copied into PDO while receiving a 1..240 Sync message. '1' means at every Sync frame, 2 at every second…
254 **255**	Data will be copied into PDO when Event occurs and is sent/executed directly

SDO 0x1800…0x1803, Transmit PDO- Communication Parameter

Defines the functionality of a PDO. Due to the fact of fixed PDO Mapping these SDOs are Read only.

Object-ID				Sub-Index			Description
0x1800	**0x1801**	**0x1802**	**0x1803**				
6	6	6	6	**0**	*U8*	*ro*	Largest sub-index supported
0xC000'0180 +Node	*0xC000'0280 +Node*	*0xC000'0380 +Node*	*0xC000'0480 +Node*	**1**	*U32*	*ro*	PDO COB-ID *Bit31 might be cleared to activate PDO*
0xFE	*0xFE*	*0xFE*	*0xFE*	**2**	*U8*	*rw*	Transmission type *Supported values: 0..240; 254*
-	-	-	-	**3**	*U16*	*ro*	Inhibit time [x100us]
-	-	-	-	**4**	*U8*	*ro*	Compatibility entry, not applicable
0 [3)]	0	0	0	**5**	*U16*	*rw*	Event timer [x1ms] *Supported values: 0; 20..65535* *0: Event off*
0	0	0	0	**6**	*U8*	*rw*	Sync start value *(TxPDO is managed when SYNC message with DLC=1 was received, SYNC data(counter) = Sync Start value, this parameter is combined w. transmission type)* *0: off, no group match*

3) this value was in previous versions represented by **SDO 0x2020:05**

Activation Example (Step by Step)

.. This Sequence shows activation of Transmit-PDO1, by clearing Bit31 .. Device must be in Pre-Operational Mode

ID	DLC	DATA (HEX)	DATA (ASCII)	Comment
0	2	80 00	..	Goto PreOperational Mode
703	1	7F	.	Heartbeat Preoperational Mode
603	8	40 00 18 01 00 00 00 00	@.....	0x1800:01 ->Read Request
583	8	43 00 18 01 83 01 00 C0	C.....	0x1800:01 0xC0000183 (3221225859/-1073741437)
603	8	23 00 18 01 83 01 00 40	#.....@	0x1800:01 0x40000183 (1073742211) — Write to 1800:01
583	8	60 00 18 01 00 00 00 00	`.....	0x1800:01 0x00000000->Write Response
603	8	40 00 18 01 00 00 00 00	@.....	0x1800:01 ->Read Request
583	8	43 00 18 01 83 01 00 40	C.....@	0x1800:01 0x40000183 (1073742211)
703	1	7F	.	
0	2	01 00	..	Goto PreOperational Mode
703	1	05	.	

.. Set Transmission Interval to 100ms

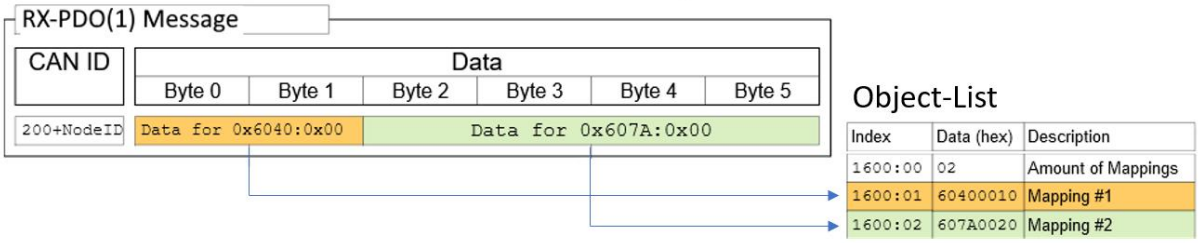
ID	DLC	DATA (HEX)	DATA (ASCII)	Comment
703	1	7F	.	
703	1	7F	.	
703	1	7F	.	
603	8	2B 00 18 05 64 00 00 00	+...d...	0x1800:05 0x0064 (100) — SDO1800:5 = 100ms
583	8	60 00 18 05 00 00 00 00	`.....	0x1800:05 0x00000000->Write Response
703	1	7F	.	
603	8	40 00 18 05 00 00 00 00	@.....	0x1800:05 ->Read Request
583	8	4B 00 18 05 64 00 00 00	K...d...	0x1800:05 0x0064 (100)
703	1	7F	.	
703	1	7F	.	
703	1	7F	.	
703	1	7F	.	Switch to OP-Mode
0	2	01 00	..	
183	8	80 41 06 00 31 03 00 00	.A..1...	PDO1-Messages
183	8	80 41 06 00 31 03 00 00	.A..1...	
183	8	80 41 06 00 31 03 00 00	.A..1...	

SDO 0x1600…0x1603 / 0x1A00…0x1A03, PDO-Mapping Parameter

Defines the Mapping to the SDO-object List of certain PDO.
Object 0x1600 defines Mapping of RxPDO1 (0x200+Node)
Object 0x1601 defines Mapping of RxPDO2 (0x300+Node)
..
Object 0x1A00 defines Mapping of TxPDO1 (0x180+Node)
Object 0x1A01 defines Mapping of TxPDO2 (0x280+Node)
..

Due to the fact of fixed PDO Mapping, this SDOs are Read-Only, means these objects will give back the static, fixed predefined Mapping information of dedicated PDOs.

Example of a Rx-PDO1 Mapping Parameter (Description)



Mapping Entry: Object-ID (2-Bye), Sub-ID (1-Byte), Number of Bits (1-Byte)

SDO 0x2010, motor drive setup

This object is used to adapt specific drive settings to motor/ gear, like max. motor current, step range etc. Corresponding knowledge is necessary. These parameters are normally preset by Adlos.

0x2010:08: Gear Ratio

Gear Ratio is predefined by ADLOS while drive end test according to drives properties. In some cases it might be useful to change Gear-Ratio by yourself, such cases might be:

- You mount the drive to spindle, and your interest is on final move on the spindle
- You mount your own gear on it

How to proceed:

- Depending on Gear-Type you need to change drive Control mode from 1:[0.1°] to 0:[um] or vice versa
  Object: SDO 0x2010-10, Setting=0 for e.g. spindle, linear moves / 1 for e.g. rotative moves
- Calculate Gear Ratio, write it into 2010-8
- Call Store CMD

Linear [um] Mode:
Slider moves 2mm/round of stepper motor axis

$$GearRatio = \frac{MoveDistance}{Motor{-}round} * 0x100'0000 = \frac{2mm}{1} * 16'777'216 = 0x200'000 = 33'554'432$$

Rotative [0.1°] Mode:
Gear output axis turns ½/round of stepper motor axisl

$$GearRatio = \frac{GearOutrounds}{Motor{-}round} * 0x100'0000 = \frac{0.5}{1} * 16'777'216 = 0x80'000 = 8'388'608$$

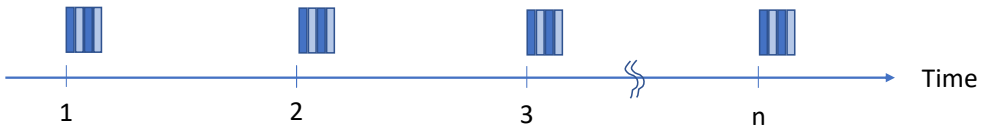
0x2010-21: Filter Control

Configuration Parameter <0x2010:21> Filter Settings Details

Filter Setting parameter enables, filter choice for each digital input, and also for analog input.

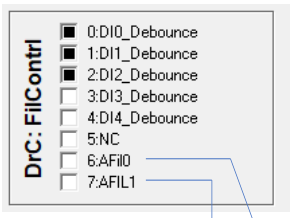
IO-Ain Sampling scheme

Every input is (if possible) sampled as an analogue input, 1 sample is converted for each channel every 1-ms



Without selecting a extra filtering by Parameter 21, every sample is added to a buffer, wich is acting as an low-pass filter of first order. Additionaly you might add a second filter by param 21, as describben below.

Filter Setting / Bit Representation



Digital Input signal Filtering
1-Bit for each Input (Set/Not Set)

Dix_Debounce	
0	Standard low-pass filter, 1. order SR=1ms t=5ms
1	Debounce filter, for mechanical switches SR=1ms t=24ms

analog Input Filter Selection

AFIL1	AFil0	Comment	
0	0	Standard low-pass filter, 1. order	SR=1ms t=5ms
0	1	Mean-Filter (of 8-Values)	SR=5ms t=40ms
1	0	Median-Filter (of 8-Values)	SR=5ms t=40ms
1	1	Reserved	

SR: sample rate

for electromechanical switches, use Debounce filter setting for proper operation.

SDO 0x2011, motor control and state

This object can be seen directly at the motor axis, that is to say these parameters are on Hardware Layer of firmware, so as an example motor axis position parameter are held in Micro-Steps. This object might be used for control or debug purpose.

SDO 0x2013, Command Execution

This object serves to execute specific motor-CMDs.

SDO 0x2014, Homing Control (Sub=1)

```
Return:     Errorcode          see error codes
```

SUB = 1: will execute Homing movement

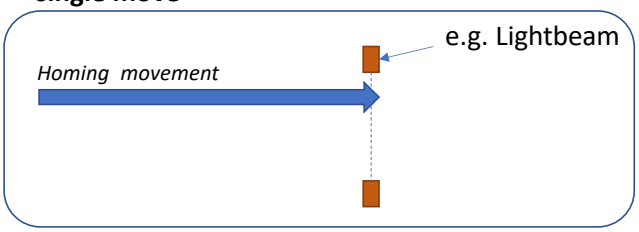
Homing modes

Mode	Type	Description	Timeout	Accuracy
0x00	Store Home	Save actual Position as Home (0)	na	na
0x01	Store 'Stall-Mark' as Home	Saves during Homing generated Mark as 0, this means actual Posiion might be different to Mark.. Use this command after a Homing on Stall, to get better precision	na	na
0x1m	Single move	Move until **stall** <m> is Torque Setting (TorqRunACCDEC * m / 15)	YES	Lo
0x2n	Single move	Move until **Input <n>, signal goes <Lo>**	YES	Med
0x3n	Single move	Move until **Input <n>, signal goes <Hi>**	YES	Med
0x4n	Double move	Move until **Input <n>, signal goes <Lo>** than direction change and 1/10 speed until **signal goes <Hi>**	YES	Hi
0x5n	Double move	Move until **Input <n>, signal goes <Hi>** than direction change and 1/8 of speed until **signal goes <Lo>**	YES	Hi
0xFF[5)]	Break/Stop	Stop Homing	na	na

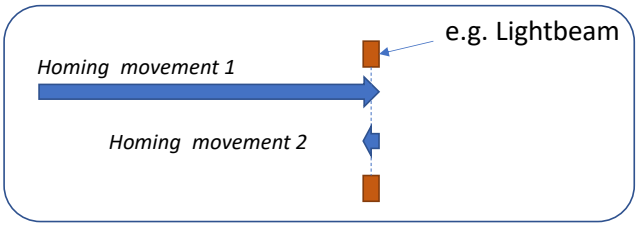
n: defines Digital Input number *m: defines Running Torque (reduction) [0..15]*

. [5)] available depending on firmware revision

single move



Double move



Homing Functionality could be also used for driving to a sensor mark, or to detect the driving boarders. Caused by this, if you would storee reached position as Zero (Homing Mark) it's needed to send a <Store-Home> frame after homing mevements.

Homing movements will take some time, you can check progress by polling the <App-state> or you Enable <PosReach> Event or SDO 2011-6 -B10.
Set Sub2 and 3 before you will start w. Sub 1.

SDO 0x2020, CAN Control Object

This object allows you to change Bitrate &, Node address and ComWatchtime. To activate changed parameters, you need to set a restart by NMT management, see Network Management.

Procedure change Bitrate/ Node Address:
- Transmit desired new node address.
- Transfer desired new bit rate.
- Store it w. 0x2013:00 if you do a (NMT->Restart Device) afterwards
- Do NMT CMD (Reset-Node (Restart Device)) or (Reset Communication (Restart of CAN Module))

… Now the new bit rate and node address are active.
… Now the device can be contacted only in this way.

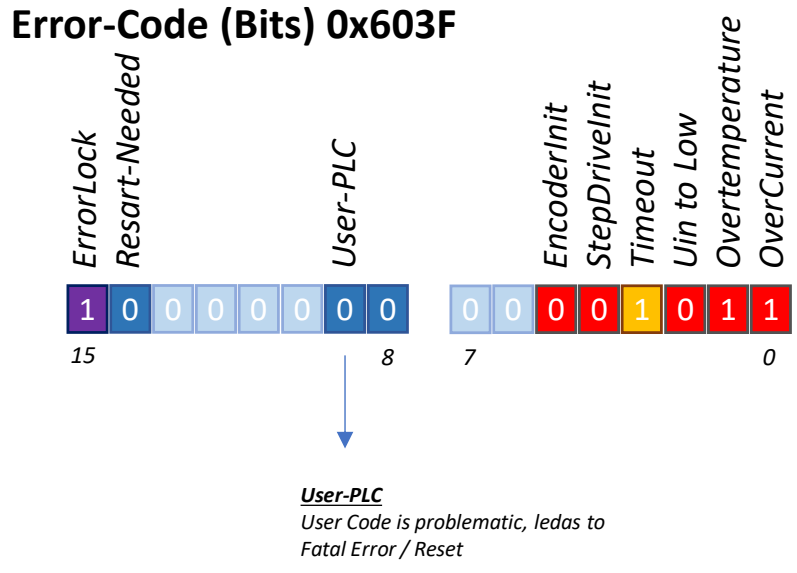
ComWatchDogTime
With this parameter it is possible to arrange that the movement of a drive is stopped, if no active communication takes place (avoiding running for ever while CAN bus is broken). Setting this time to a longer value than you do cyclic/periodically refresh your driving command. So if you communicate all 100ms w. this device you should set this value to >100ms, e.g. 200ms.

ComWatchDogTime[2] ms	U16	0	rw	YES	0: off / 50…65535 ms

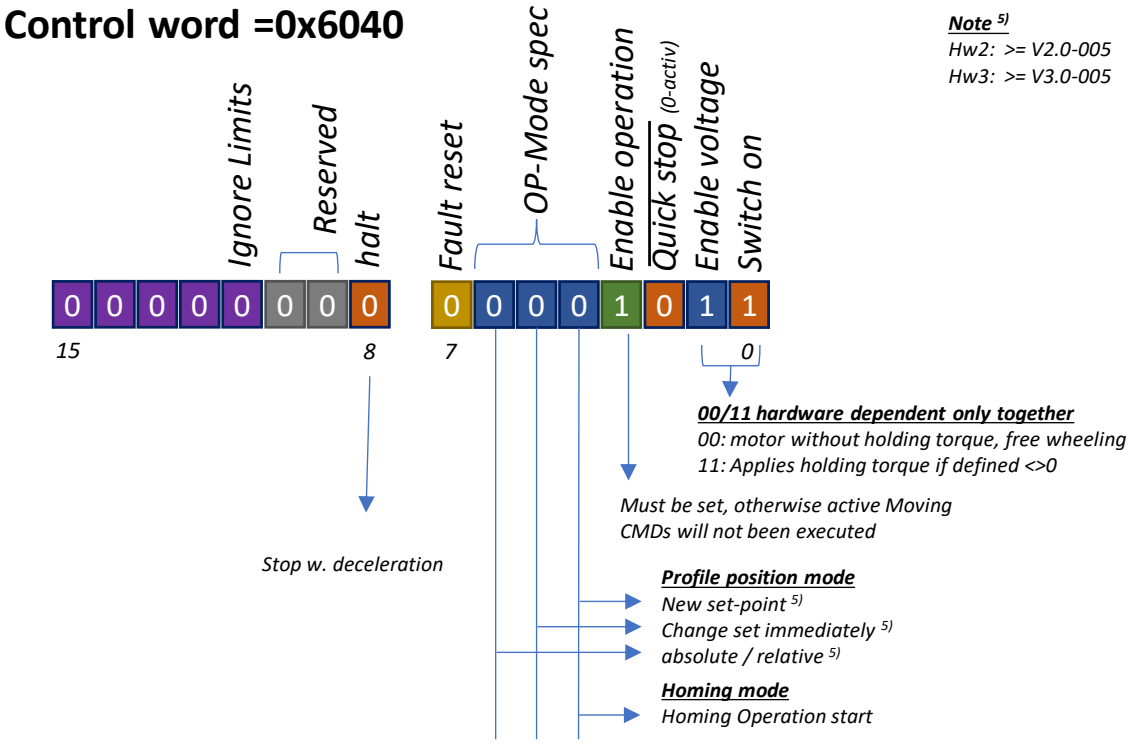
If you e.g have sent a rotate command, the drive turns at this speed. If contact is broken for some reason (PC off) the drive does not stop. But if you set this parameter to 2000ms, the motors stops 2000ms after the last communication CMD was received.

Control bits
Bit0 NMT change automatically at Startup
Reset: NMT Operation state is changing automatically at startup to <PreOperational State>
Set: NMT Operation state is changing automatically at startup to <Operational State>
Bit4 Closed Loop
Reset motor position is not controlled while passive
Set motor position is controlled to target and 0x6067, position might be corrected automatiocally

SDO 0x603F, ErrorCode



SDO 0x6040, Controlword



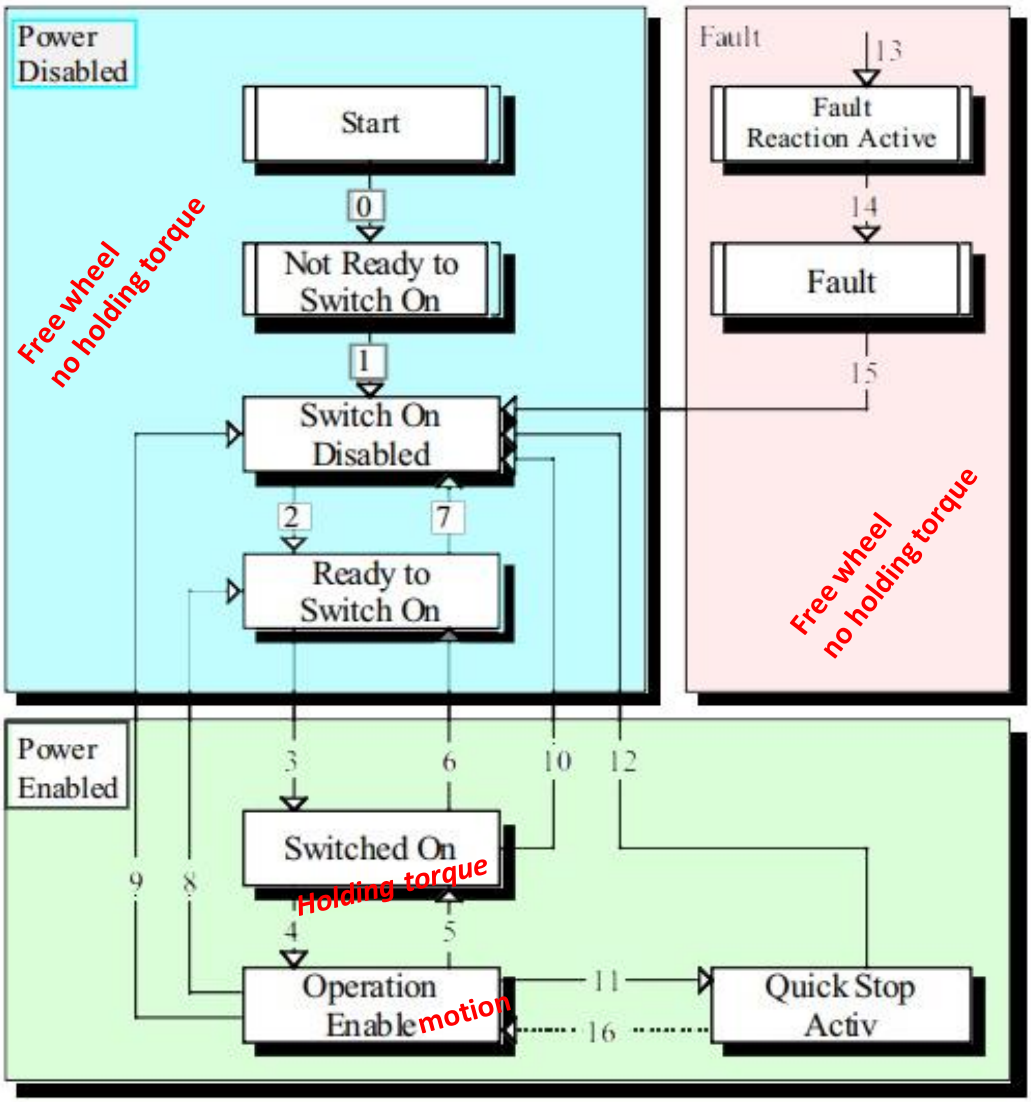
BITS 0 – 3 AND 7:

Device control commands are triggered by the following bit patterns in the *controlword*:

Command	Bit of the *controlword*					Transitions
	Fault reset	Enable operation	Quick stop	Enable voltage	Switch on	
Shutdown	0	X	1	1	0	2,6,8
Switch on	0	0	1	1	1	3*
Switch on	0	1	1	1	1	3**
Disable voltage	0	X	X	0	X	7,9,10,12
Quick stop	0	X	0	1	X	7,10,11
Disable operation	0	0	1	1	1	5
Enable operation	0	1	1	1	1	4,16
Fault reset	↑	X	X	X	X	15

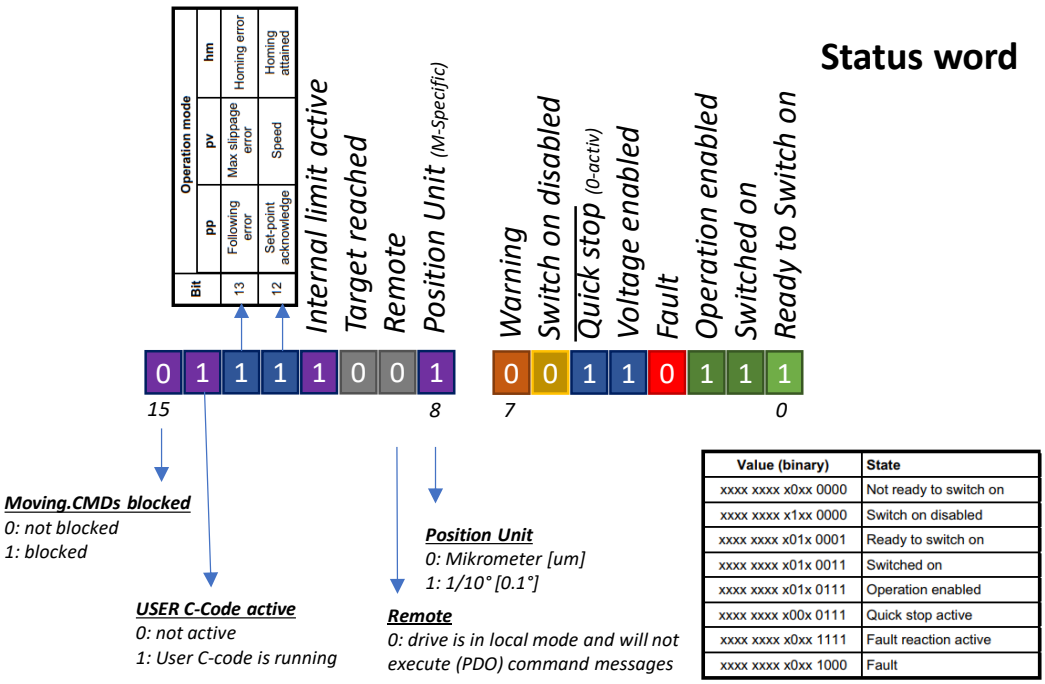
Table 4: Device control commands **(bits marked X are irrelevant, * ... In the state SWITCHED ON the drive executes the functionality of this state., ** ... It exists no functionality in the state SWITCHED ON. The drive does not do any in this state.)**

State-machine according DSP402



- Transitions (0,1,2,3,4, 6, 8, 9, 10, 11, 12) might be done automatically by drive
- Transitions (11, 15) are introduced by 0x6040 Control word

SDO 0x6041, Statusword



Value (binary)	State
xxxx xxxx x0xx 0000	Not ready to switch on
xxxx xxxx x1xx 0000	Switch on disabled
xxxx xxxx x01x 0001	Ready to switch on
xxxx xxxx x01x 0011	Switched on
xxxx xxxx x01x 0111	Operation enabled
xxxx xxxx x00x 0111	Quick stop active
xxxx xxxx x0xx 1111	Fault reaction active
xxxx xxxx x0xx 1000	Fault

Notes:

Moving CMDs blocked

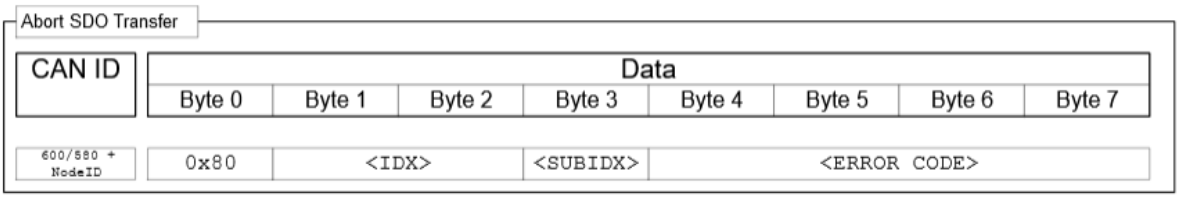
- CAN Moving CMDs, where the drive needs to do some movements are not executed. (like GotoPosition/Rotate/Homing)… this might be as an impact of internally running user c-sequence, this bit might also been set if a drive parameter change needs a Restart for proper Recalculation, see ErrorCode for mor information

SDO 0x607D, Limits

- Limits are newly set after the homing run.

SDO error messages

A SDO request always will be responded, that is to say a WR request as well. But the SDO request also could report a error code.



Example:

Verworfene / nicht zuweisbare Datenpackete [0x60 = Write Response] [0x80 = Fehler]

Msg.-Type	ID	DLC	DATA (HEX)	DATA (ASCII)
STD	583	8	60 11 20 02 00 00 00 00	`. .....
STD	583	8	80 10 20 07 FF 00 00 00	.. ..

Illustration 1: not assigned request by the tool

The second answer, which could not be assigned by the tool, concerns a error message.
The error message is coded as follows:

- 1. Byte 0x80
- 2.&3. Byte SDO Object in this case 0x2010 (Liddle Endian coded)
- 4. Byte errror message in this case 0xFF = (-1)

Official CANopen Error codes

Error Code	Description
05030000h	*toggle bit not changed*: Gültig nur bei "normal transfer" oder "block transfer". Das Bit, welches nach jeder Übertragung zu alternieren hat, hat seinen Zustand nicht geändert.
05040001h	*command specifier unknown*: Das Byte 0 des Datenblocks enthielt einen nicht zulässigen Befehl.
06010000h	*unsupported access*: Falls über CAN over EtherCAT (CoE) ein "complete access" angefordert wurde (wird nicht unterstützt.)
06010002h	*read only entry*: Es wurde versucht, auf ein konstantes oder nur lesbares Objekt zu schreiben.
06020000h	*object not existing*: Es wurde versucht, auf ein nicht vorhandenes Objekt zu zugreifen (Index fehlerhaft).
06040041h	*objekt cannot be pdo mapped*: Es wurde versucht, ein Objekt in das PDO zu mappen, für dass das nicht zulässig ist.
06040042h	*mapped pdo exceed pdo*: Würde das gewünschte Objekt in das PDO-Mapping angehängt werden, würden die 8Byte des PDO-Mappings überschritten.
06070012h	*parameter length too long*: Es wurde versucht, auf ein Objekt mit zu vielen Daten zu schreiben; zum Beispiel mit <CMD>=23h (4 Byte) auf ein Objekt des Types Unsigned8, korrekt wäre das <CMD>=2Fh.
06070013h	*parameter length too short*: Es wurde versucht, auf ein Objekt mit zu wenig Daten zu schreiben; zum Beispiel mit <CMD>=2Fh (1 Byte) auf ein Objekt des Types Unsigned32, korrekt wäre das <CMD>=23h.
06090011h	*subindex not existing*: Es wurde versucht, auf ein ungültiges Subindex eines Objektes zu zugreifen, der Index hingegen würde existieren.

Error Code	Description
06090031h	*value too great*: Einige Objekte unterliegen Restriktionen in der Größe des Wertes, in diesem Fall wurde versucht, einen zu hohen Wert in das Objekt zu schreiben. Zum Beispiel darf das Objekt "Pre-defined error field: Number of errors" bei 1003h:00 nur auf den Wert "0" gesetzt werden, alle anderen Zahlenwerte provozieren diesen Fehler.
06090032h	*value too small*: Einige Objekte unterliegen Restriktionen in der Größe des Wertes. In diesem Fall wurde versucht, einen zu niedrigen Wert in das Objekt zu schreiben.
08000000h	*general error*: Allgemeiner Fehler, der in keine andere Kategorie passt.
08000022h	*data cannot be read or stored in this state*: Die Parameter des PDOs dürfen nur im State *Stopped* oder "Pre-Operational" verändert werden. Ein Schreibzugriff auf die Objekte 1400h bis 1407h, 1600h bis 1607h, 1800h bis 1807h und 1A00h bis 1A07h ist im Zustand "Operational" nicht zulässig.

Kannmotion extra Error codes, used only in some cases

```
0:      eMS_OK                          //!< no error Fehler
-1:     eMS_ERR_OUTofRange              //!< error, Parameter are outside the valid range
-2:     eMS_ERR_ParamisWrProtected      //!< error, Parameter can't be written
-3:     eMS_ERR_CMDnotAccepted          //!< error, Command can't be executed, da for example motor is busy
-4:     eMS_ERR_CMDnotKnown             //!< error, Command unknown
-5:     eMS_ERR_ParamisNotKnown         //!< error, Parameter is unknown
```

Translation of the right column:

0: no error
-1: error, parameter outside valid range
-2: error, parameter cannot be written
-3: error, command could not be executed, because e.g. still in progress
-4: error, command unknown
-5: error, parameter unknown

5. Implemented PDO's (Process Data Objects)

Process data are transmitted by PDO-telegram. In comparison to the SDO-telegram this is done with a higher priority and the telegram doesn`t require an answer. The 8 data bytes are freely available and can be used as pure user data. Usually the user data are mapped with objects in the SDO object list.
PDO-communication can be executed cyclical at the request of the master or event based. The event-based variant is used, if it has been activated by SDO!

PDO-Object list

RX-PDOs

PDO	ID	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	DLC
1	0x200+NodeID	**0x607A:00** Target Position				-	-	-	-	4[1)]
						0x6040:00 Control word		-	-	6..8
2	0x300+NodeID	**0x2013:00** CMD	0	**0x2011:04** Target Speed		-	-	-	-	4..8
3	0x400+NodeID	**0x6081:00** Profile velocity (max. Speed at PP)				-	-	-	-	4..8
4	0x500+NodeID	*CUSTOMIZED, mapped into USER specific Code region, see also App-Note 100631*								

Table 1: PDOs RX Motor

TX-PDOs

PDO	ID	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	DLC
1	0x180+NodeID	**0x6064:00** Actual Position				**0x6041:00** Status word		**0x6044:00** Velocity control (actual value)		8
2	0x280+NodeID	**0x603F:00** Error-Code		**0x1001:00** Error Reg	**0x2011:01** MotorState	**0x2011:08** Temperature				5
3	0x380+NodeID									
4	0x480+NodeID	*CUSTOMIZED, mapped into USER specific Code region, see also App-Note 100631*								

Table 2: PDOs Tx Motor

Notes:

[1)] For backward compatibility w. older Releases

Details PDO4

PDO4 Rx and TX are mapped into User specific App-Code Area. Caused by this you are enabled to customize your drive by using KannMOTION customizing approach. By writing some own ANSI C-code lines you are able to integrate your own behave on PDO4 Rx. With PDO4-Tx you can send your individual data on CANopen.

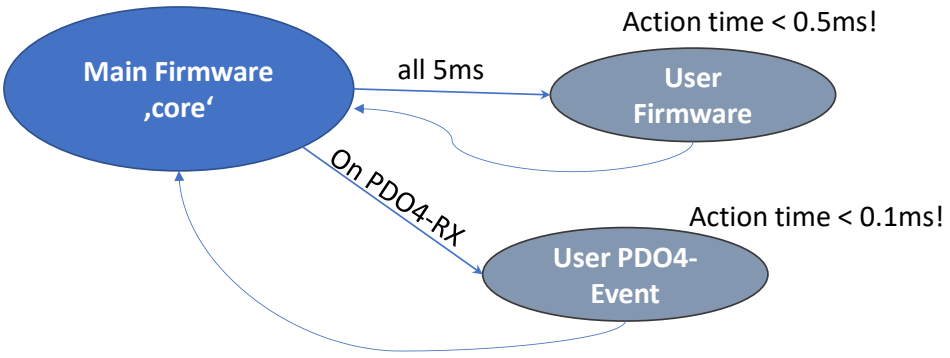


Figure 1: Basic User Section Call

Example code:

Your code interpreter is written into Event-Handler of PDO4-Rx. This example takes first byte from PDO-4 as command information. On Command=0x02 an Answer will be sent through PDO4-Tx.

```
/*****************************************************************************************/
/*!
 *  \brief     SPS-USER Function Event is called at CANopen-PDO4 Rx
 *  \details
*   \param    pRxData, Pointer to Data received
*   \param    Datalength, Data count [0..8]
*   \return
*****************************************************************************************/
void LOCATEUSER AppCSPS_USER_SEQ_CAN_RX_Event(tCANDATA* pRxData, UI_8 Datalength)
{
   // .. My Command interpreter
   switch (pRxData->u08_Data[0])
   {
     // CMD=0: Goto-Pos, Target Position is in
     case 00:
     {
        if (Datalength!=8)
        {
          return;
        }
        // Call Goto Function
        if (stAppCSPS.SPSCallFunctions.GotoFuncP((SI_32)pRxData->i32_Data[1],eGOTO_um_01deg)==eMS_OK)
        {
          stAppCSPS.SPSUserVar.u16_Timer5ms[0]=800;              // 4s -> 5ms x 400
          stAppCSPS.SPSUserVar.u8_StepChain[0]=4;                // next Step = Delay
        }
      break;
     }
     // CMD=2: Query...
     case 02:
     {
       tCANDATA CanTxData;
       CanTxData.u32_Data[0]=0x01234567;
       CanTxData.u32_Data[1]=0x89ABCDEF;
       if (stAppCSPS.SPSCallFunctions.CANopen_PDO4_Send(&CanTxData,8))
       {
       }
      break;
     }
   }
}
```

6. Appendix

References and extracts from the standard profiles

CAN-CANopen sources

The original CAN-specification of Bosch Ltd. (public):
http://www.bosch-semiconductors.de/media/ubk_semiconductors/pdf_1/canliteratur/can2spec.pdf

Further links:
https://www.nxp.com/docs/en/reference-manual/BCANPSV2.pdf
https://en.wikipedia.org/wiki/Unified_Diagnostic_Services

Useful links on the topic of CANopen
(in this document occasionally data from these sources are used):
http://www.leuze.com/media/assets/archive/TD_canopen_guideline_de.pdf
http://www.microcontrol.net/produkte/protokollstacks/canopen/
http://www.microcontrol.net/download/appnotes/canopen_intro.pdf
https://www.can-cia.org/standardization/specifications
http://www.canopensolutions.com/english/about_canopen/about_canopen.shtml
http://www.feldbusse.de/CanOpen/komprofil.shtml

Profile DS301

In the DS301-protocol only fundamental requirements (group A) are put into practice.

Index	Object	Name	Type	Attr.	Mandatory						
					A	B	C	D	E	F	G
1000	VAR	device type	UNSIGNED32	ro	M						
1001	VAR	error register	UNSIGNED8	ro	M						
1018	RECORD	Identity Object	Identity (23h)	ro	M						
1201	RECORD	2nd Server SDO parameter	SDO Parameter (22h)	rw		M**					
...	...	...	...	...		...					
127F	RECORD	128th Server SDO parameter	SDO Parameter (22h)	rw		M**					
1280	RECORD	1st Client SDO parameter	SDO Parameter (22h)	rw			M**				
...	...	...	...	...			...				
12FF	RECORD	128th Client SDO parameter	SDO Parameter (22h)	rw			M**				
1400	RECORD	1st receive PDO Parameter	PDO CommPar (20h)	rw				M*			
...	...	...	...	...				...			
15FF	RECORD	512th receive PDO Parameter	PDO CommPar (20h)	rw				M*			
1600	RECORD	1st receive PDO Mapping	PDO Mapping (21h)	rw					M*		
...	...	...	...	...					...		
17FF	RECORD	512th receive PDO Mapping	PDO Mapping (21h)	rw					M*		
1800	RECORD	1st transmit PDO Parameter	PDO CommPar (20h)	rw						M*	
...	...	...	...	...						...	
19FF	RECORD	512th transmit PDO Parameter	PDO CommPar (20h)	rw						M*	
1A00	RECORD	1st transmit PDO Mapping	PDO Mapping (21h)	rw							M*
...	...	...	...	...							...
1BFF	RECORD	512th transmit PDO Mapping	PDO Mapping (21h)	rw							M*

Table 1: Requirements and realization of the DS301 protocol

Captions	
*	Only if PDO's are supported.
**	Only if SDO's are supported.
A	Basic requirements
B	Server SDO Parameters
C	Client SDO Parameters
D	Receive PDO Communication Parameter
E	Receive PDO Mapping Parameter
F	Transmit PDO Communication Parameter
G	Transmit PDO Mapping Parameter

Profile DSP402

At DSP402 the basic objects and those of the "Position control function" ("pc") are put into practice.

Index	Object	Name	Type	Attr.	andatory / Optional / Condition							
					all	hm	pp	pc	ip	pv	tq	vl
6040	VAR	Controlword	UNSIGNED16	rw	M							
6041	VAR	Statusword	UNSIGNED16	ro	M							
6060	VAR	Modes of operation	INTEGER8	rw	M							
6061	VAR	Modes of operation display	INTEGER8	ro	M							
607A	VAR	Target position	INTEGER32	rw			M	M				
6081	VAR	Profile velocity	UNSIGNED32	rw			M			M		
6083	VAR	Profile acceleration	UNSIGNED32	rw			M			M		
6084	VAR	Profile deceleration	UNSIGNED32	rw			M			M		
6086	VAR	Motion profile type	INTEGER16	rw			M			M		
6098	VAR	Homing method	INTEGER8	rw		M						
6099	ARRAY	Homing speeds	UNSIGNED32	rw		M						
6064	VAR	Position actual value	INTEGER32	ro				M				
6069	VAR	Velocity sensor actual value	INTEGER32	ro						M		
606B	VAR	Velocity demand value	INTEGER32	ro						M		
606C	VAR	Velocity actual value	INTEGER32	ro						M		
60FF	VAR	Target velocity	INTEGER32	rw						M		
6071	VAR	Target torque	INTEGER16	rw							M	
6087	VAR	Torque slope	UNSIGNED32	rw							M	
6088	VAR	Torque profile type	INTEGER16	rw							M	
6042	VAR	vl target velocity	INTEGER16	rw								M
6043	VAR	vl velocity demand	INTEGER16	ro								M
6044	VAR	vl control effort	INTEGER16	ro								M
6046	ARRAY	vl velocity min max amount	UNSIGNED32	rw								M
6048	RECORD	vl velocity acceleration	vl velocity acceleration deceleration record	rw								M
6049	RECORD	vl velocity deceleration	vl velocity acceleration deceleration record	rw								M
607D	VAR	software_position_limits	Min/MaxLimit	r								

Table 2: Requirements and realization of the DSP402-protocol

Captions	
all	Is crucial
hm	Homing mode
pp	Profile position mode
pc	Position control function
ip	Interpolated position mode
pv	Profile velocity mode
tq	Profile torque mode
vl	Velocity mode

Timing

The transmission time of PDO- or a SDO-package depends on the length of the package and the adjusted baud rate. The structure of a package can be seen in illustration 3: The complete frame including maximally 8 data bytes amounts 108 bit.

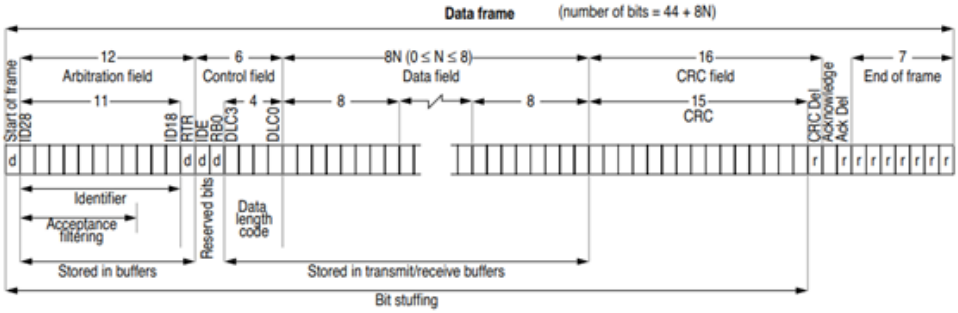


Illustration 2: structure of data frame

The typical transmission time can be calculated from the data frame and the baud rate:

Parameter	Description	Wert
Bitzeit	**250k**	4 us/Bit
Data frame	Complete PDO data frame with 8 data bytes (44+64)	108 Bit
Transmission time frame	108 Bit * 4 us/Bit	432 us
Bus Idle min	3 * 4us	12 us
Maximal frame size	**Incl. min. IDLE time**	**444 us**

Table 3: Transmission time data frame

For the calculation you calculate with the maximal number of 8 data bytes, although the length of the PDO-packages is dynamic and in most cases smaller.

Bus usage

For the execution of the main application positioning movement HMI sends the PDO-package. The resulting bus allocation can be seen in the following table:

Pos	Funktion	Richtung	Paket	Dauer [us]
1	Drive comand Motor	HMI -> SmartMotor	PDO1	216
2	Motor Event State change / Moving	SmartMotor ->HMI	PDO1	216
3	Motor Event State change / auf Position	SmartMotor ->HMI	PDO1	216
			Total	**648 us**

Table 4: Timing Standard-Application

Important information:

- By default, selection of the identifiers the PDOs pass always in priority, that is to say SDO-accesses always have a lower priority on the bus.
- The message events (on the PDOs as well) have higher priority as the transmit-PDOs of the HMI.
- Heartbeat has the lowest bus priority.

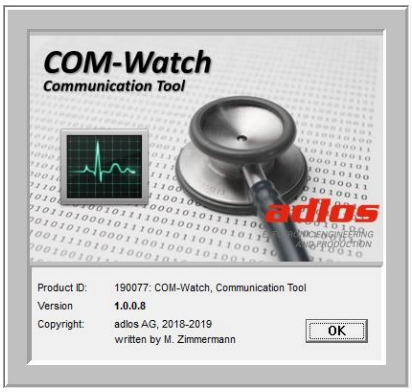
1 Adlos Win32-APPs

KannMotion Manager tool (190081), manage your drives

KannMOTION Manager is the general tool for our generation 2 (GEN2) drives. This tool comes with an integrated C-coder and a visual drag and drop user interface for customizing your drive.

https://kannmotion.adlos.com/download/kannmotionmanager/application/SetupKannMOTIONManager.zip

ComWatch Communication Tool (190077)



ComWatch might be used with this drives at very seldom case.
To use Comwatch, you need 100550 Connection cable and it's necessary to open the driver housing to connect to internal PCB directly.

https://kannmotion.adlos.com/download/comwatchtool/ComWatchSetup.zip

KannMOTION API

Adlos offers a windows API (Library) to communicate with our drives. The API enables much shorter implementation of KannMOTION communication with your own Windows based toolset and application.

Part number	Short / level	Description	
190073	LEVEL1 API-LLL	Low Level Abstraction offers RD/WR functions to Com, organizes Checksum and protocol Itself	
190074	LEVEL2 API-HAL	Hardware abstraction offers data object modeling, means it will take care bout device specific XML-files	
190080	LEVEL3 API-BAL	Bus abstraction Offers bus data support like CAN	