ESTUN Servo CANopen User's Manual

（Version：V1.04）

ESTUN AUTOMATION TECHNOLOGY CO., LTD

— Total Solution Supplier

——Contents ——

Chapter 1 Brief introduction

1.1 CAN main files

Document Name	Source
CiA DS 301 V 4.01: CANopen Communication Profile for Industrial Systems - based on CAL	**CiA**
CiA DSP 402 V 2.0: CANopen Device Profile	**CiA**

1.2 Terms and abbreviations

CAN	Controller Area Network
CiA	CAN in Automation International Users and Manufacturers Group.
COB	Communication Object (CAN message). A unit of transportation in a CAN network. Data must be sent across a network inside a COB. The COB itself is part of the CAN message frame.
EDS	Electronic Data Sheet. A node-specific ASCII-format file required when configuring the CAN network.The EDS file contains general information on the node and its dictionary objects (parameters).
LMT	Layer Management. One of the service elements of the CAN Application Layer in the CAN Reference Model. It serves to configure parameters for each layer in the CAN Reference Model.
NMT	Network Management. One of the service elements of the CAN Application Layer in the CAN Reference Model. It performs initialization, configuration and error handling on a CAN network.
OD	A local storage of all Communication Objects (COB) recognized by a device.
Parameter	A parameter is an operating instruction for the drive. Parameters can be read and programmed with the drive control panel.
PDO	Process Data Object; a type of COB. Used for transmitting time-critical data, such as control commands, references and actual values.
RO	Denotes read-only access.
RW	Denotes read/write access.
SDO	Service Data Object; a type of COB. Used for transmitting non-time critical data, such as parameters.

1.3 Brief introduction

CANopen is a higher-layer protocol based on the CAN (Control Area Network) serial bus system and the CAL (CAN Application Layer). CANopen assumes that the hardware of the connected device has a CAN transceiver and a CAN controller as specified in ISO 11898.

The CANopen Communication Profile, CiA DS-301, includes both cyclic and event-driven communication, which makes it possible to reduce the bus load to minimum while still maintaining extremely short reaction times. High communication performance can be achieved at relatively low baud rates, thus reducing EMC problems and cable costs.

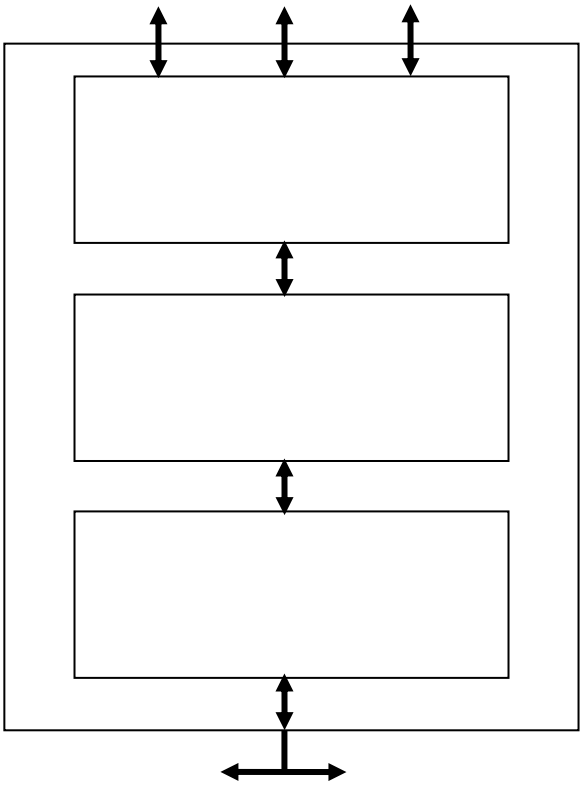
CANopen device profiles define both direct access to drive parameter and time-critical process data communication. The NCAN-02 fulfils CiA (CAN in Automation) standard DSP-402 (Drives and Motion Control), supporting the 'Manufacturer Specific' operating mode only.

The physical medium of CANopen is a differentially-driven two-wire bus line with common return according to ISO 11898. The maximum length of the bus is limited by the communication speed as follows:

Baud Rate	Max. Bus Length
1M bit/s	25 m
500k bit/s	100 m
250k bit/s	250 m
125k bit/s	500 m
100k bit/s	600 m
50k bit/s	1000 m

The maximum theoretical number of nodes is 127. However, in practice, the maximum number depends on the capabilities of the CAN transceivers used.Each node is an independent CANopen device, which contains maximum 8 logical devices.And one logical device controls one axis for servo drive.

The model of CANopen device is as below:



1、Communication interface

Communication interface defines PDO, SDO, NMT and other protocol about message transmission.

2、Object Dictionary

The key concept of CANopen is device Object Dictionary,which is used in other live bus （Profibus，Interbus-S）. CANopen can visit all the parameters of the servo drive by OD.Notes: OD is not a part of CAL, but relised in CANopen.

The object of OD can be recognised by a 16 bit index and a 8 bit index (please refer to **Appendix Object dictionary**). OD is classified by CANopen as below:

Main Index	Object	Main Index	Object
0000H	Reserved	2000H-5FFFH	Manufacturer set object
0001H-001FH	Static data type	6000H-67FFH	Logic device 1 object
0020H-003FH	Complex data type	6800H-6FFFH	Logic device 2 object
0040H-005FH	Manufacturer set data type	7000H-77FFH	Logic device 3 object
0060-007FH	Device standard basic data type	…	…
0080H-009FH	Device standard complex data type	9800H-9FFFH	Logic device 8 object
00A0H-0FFFH	Reserved	A000H-BFFFH	Standard interface object
1000H-1FFFH	Communication standard	C000H-FFFFH	Reserved

The most commonly used object:

1）1000H-1FFFH，communication standard object,to realize network configuration and PDO mapping,etc.

2）2000H-5FFFH, manufacturer specified object, to realize read and write Pn xxx parameters.

3）6000H-9FFFH, Logic device 1-8 object, which belong to CIA 402 device sub-protocol. It is possible to operate

these objects for carry out a variety of servo drive modes. Each logic device is related to one axis, so, ProNet can index the objects in the range of 6000H-67FFH, and the EDS drive can index the objects in the range of 6000H-6FFFH, and that the ETS can index the objects in range of 6000H-9FFFH. To get the corresponding objects of the second-axis and the third-axis, separately shifted the objects of first axis to 0800H and 1000H. For example, the control word of the first-axis which of the index is 6040H, well than, the control words for the second-axis and the third-axis are 6840H and 7040H respectively. Also, it is possible to find the index of the corresponding objects by the object dictionary in the appendix. For the operational mode, this manual only takes the using of the first-axis as the example. The operation of the second-axis and the third-axis are the same with it for the EDS and the ETS, so it is unnecessary to go into details.

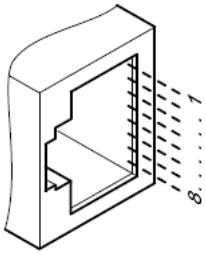
3、Application

It controls parameters of servo drive and serval other assistant functions by serval operation mode. ProNet/EDS/ETS currently supports 4 control modes: profile position mode、profile velocity mode、profile torque mode、interpolation position mode and homing mode.

Further information can be obtained from the CAN in Automation International Users and Manufacturers Group (*www.can-cia.de*).

Chapter 2 Cabling and wiring

●The layout of CN3 terminal

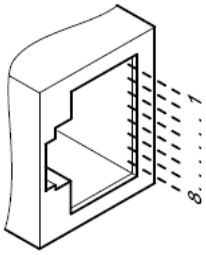


Pin number	Name	Function
1	—	Reserved
2	—	
3	485+	RS-485 communication terminal
4	ISO_GND	Isolated GND
5	ISO_GND	
6	485-	RS-485 communication terminal
7	CANH	CAN communication terminal
8	CANL	CAN communication terminal

Note:

1. Do not short terminal 1 and 2 of CN3.
2. If connecting more than 16 CAN nodes,please contact ESTUN customer service.

● The layout of CN4 terminal

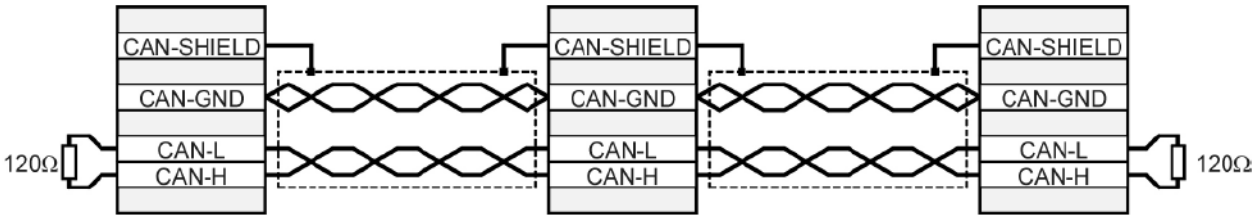


Pin number	Name	Function
1	—	Reserved
2	—	
3	485+	RS-485 communication terminal
4	ISO_GND	Isolated GND
5	ISO_GND	
6	485-	RS-485 communication terminal
7	CANH	CAN communication terminal
8	CANL	CAN communication terminal

CN3 is always the input terminal of communication cable and CN4 is always the output terminal of communication cable. (If connection to another communication node is necessary, the cable will connect CN4 to next communication node. If not, a terminal resistor could be applied at CN4). When multiple ProNet devices are connected, it is forbidden to connect the CN3 terminals of different drives directly.

For example, a network is composed of one PLC, three ProNet drives called A, B and C. The cabling network is as below: PLC → CN3 of drive A，CN4 of drive A → CN3 of drive B，CN4 of drive B → CN3 of drive C, CN4 of drive C → 120Ω resistor.

The two ends of the CAN cable have to be terminated by a resistor of 120Ω (1%, 1/4W) as below.



Please select the bus cable with double twisted pair cables and shielding layer, one pair for connecting CAN-L and CAN-H, another pair for grounding.

Chapter 3 CANopen communication

CAL supplies all network management service and message transferring protocol with defining the content of object or type of object for communication. It defines how instead of what, which is the strength of CANopen.

CANopen is developed based on CAL. It applies CAL protocol subsets for communication and service and creates a solution to DCS. CANopen could freely extend the node function to simplicity or complex while the network nodes are accessible and available to each other.

The key concept of CANopen is object dictionary. This way of object description is also applied to other fieldbus system like Profibus and Interbus-S. CANopen communication could access to all the parameter of drivers through object dictionary. Please notice object dictionary is not one part of CAL, instead of which it is realized in CANopen.

CANopen communication defines several types of objects as below…

Abbreviation	Full Spell	Description
SDO	Service Data Object	Used for normal parameterization of the servo controller
PDO	Process Data Object	Fast exchange of process data (e.g. velocity actual value) possible.
SYNC	Synchronization Message	Synchronization of several CAN nodes
EMCY	Emergency Message	Used to transmit error messages of the servo controller.
NMT	Network Management	Used for network services. For example usercan act on all controllers at the same time via this object type.
Heartbeat	Error Control Protocol	Used for observing all nodes by cyclic messages.

CAN employs data frames for transferring data between the host (controller) and the nodes on the bus. The following figure presents the structure of the data frame.

Start of frame	Arbitration field		Control field	Date field	Cyclical redundancy check	Acknowledged field	End of frame
	COB-ID	RTR					
1BIT	1 OR 29 BITS	1BIT	6BITS	0～8BYTES	16BITS	2BITS	7BITS

Our drivers doesn't support remote frame currently. The detail of COB-ID is as below.

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

3.1 CAN identifier list

Object	COB-ID bit10~7 （binary）	COB-ID (hex)	Index in OD
NMT	0000	000_h	—
SYNC	0001	080_h	1005_h、1006_h、1007_h
TIME STAMP	0010	100_h	1012_h、1013_h
EMCY	0001	081_h ~ $0FF_h$	1024_h、1015_h
PDO1（transimit）	0011	181_h ~ $1FF_h$	1800_h
PDO1（receive）	0100	201_h ~ $27F_h$	1400_h
PDO2（transimit）	0101	281_h ~ $2FF_h$	1801_h
PDO2（receive）	0110	301_h ~ $37F_h$	1401_h
PDO3（transimit）	0111	381_h ~ $3FF_h$	1802_h
PDO3（receive）	1000	401_h ~ $47F_h$	1402_h
PDO4（transimit）	1001	481_h ~ $4FF_h$	1803_h
PDO4（receive）	1010	501_h ~ $57F_h$	1403_h
SDO（transimit）	1011	581_h ~ $5FF_h$	1200_h
SDO（receive）	1100	601_h ~ $67F_h$	1200_h
Heartbeat	1110	701_h ~ $77F_h$	1016_h、1017_h

Note:

1. PDO/SDO's send/receive is observed by (slave) CAN.
2. Our drive's CANopen protocol currently supports 4 transimit PDO and 4 receive PDO.

3.2 SDO

SDO is used to visit the object dictionary of a device. Visitor is called client. The CANopen device whose object dictionary is visited and required to supply the asked service is called server. CANopen messages from a client and servo all contain 8 bits (Not all of them are meaningful). A request from a client must be confirmed by a server

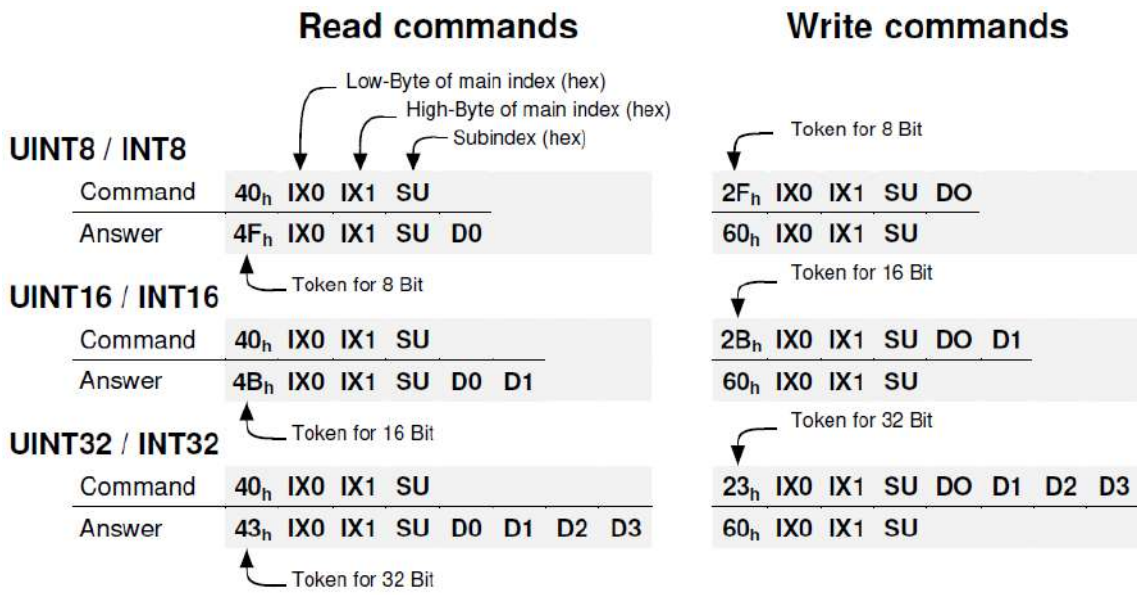
There are 2 method of conveying SDO:

- Expedited transfer：contains 4 bytes at maximum
- Segmented transfer：contains more than 4 bytes

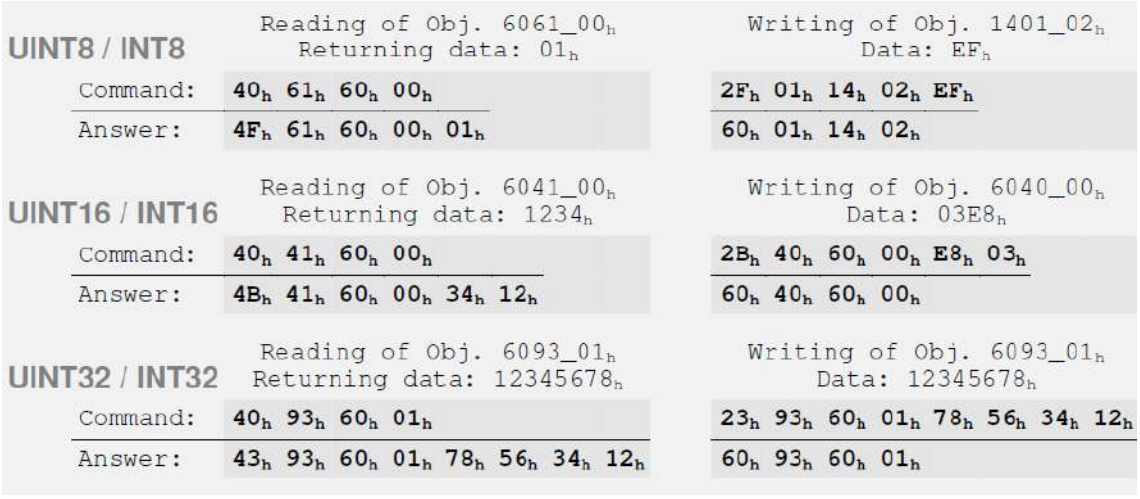
Basic structure of SDO：

Byte0	Byte1～2	Byte3	Byte4～7
SDO	Object reference	Sub-object reference	data

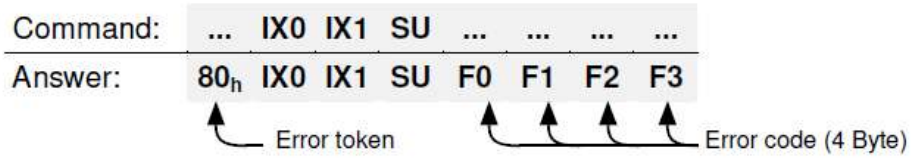
SDO read/write command structure:



Example:



SDO-error messages：



Error code F3 F2 F1 F0	Description
05 03 00 00$_h$	Toggle bit not alternated
05 04 00 01$_h$	Client / server command specifier not valid or unknown
06 01 00 00$_h$	Unsupported access to an object
06 01 00 01$_h$	Attempt to read a write only object
06 01 00 02$_h$	Attempt to write a read only object
06 02 00 00$_h$	Object does not exist in the object dictionary
06 04 00 41$_h$	Object cannot be mapped to the PDO
06 04 00 42$_h$	The number and length of the objects to be mapped would exceed PDO length
06 04 00 47$_h$	General internal incompatibility in the device
06 07 00 10$_h$	Data type does not match, length of service parameter does not match
06 07 00 12$_h$	Data type does not match, length of service parameter too high
06 07 00 13$_h$	Data type does not match, length of service parameter too low
06 09 00 11$_h$	Sub-index does not exist
06 04 00 43$_h$	General parameter incompatibility
06 06 00 00$_h$	Access failed due to an hardware error *1)
06 09 00 30$_h$	Value range of parameter exceeded
06 09 00 31$_h$	Value of parameter written too high
06 09 00 32$_h$	Value of parameter written too low
06 09 00 36$_h$	Maximum value is less than minimum value
08 00 00 20$_h$	Data cannot be transferred or stored to the application *1)
08 00 00 21$_h$	Data cannot be transferred or stored to the application because of local control
08 00 00 22$_h$	Data cannot be transferred or stored to the application because of the present device state *3)
08 00 00 23$_h$	No Object Dictionary is present *2)

3.3 PDO

PDO is applied to transferring real time data which will be conveyed from a producer to one or multiple clients. Data transferring will be limited to 1 to 8 bytes. There is no hand-shake restriction in PDO communication, which means data has been redefined, so clients could process the received data for vary short time. PDO content will be only defined by its CAN ID, assuming producers and clients know PDO content from its CAN ID.

2 objects in object dictionary are used for each PDO.

- PDO communication parameter：It contains COB-ID, transferring type, restriction time and cycle of timer used by PDO.

- PDO mapping parameter：It contains a list of objects in the object dictionary. These objects are mapped into PDO, includes their data length in bits. Producers and clients must know this mapping to explain the content of PDO.

The content of PDO's message is predefined or configured when the network initializes. Mapping application object into PDO is described in object dictionary. If a device (producer and client) support dynamic mapping, SDO could be used to configure PDO's mapping parameter. Our servo drive supports dynamic PDO mapping. There are 2 rules for PDO mapping to follow..

1. Each PDO could be mapped into 4 objects.
2. The length of each PDO will be no more than 64 bits.

PDO mapping process：

1. Set the sub-index of PDO coordinated mapping parameter ($1600_{h, 1601\,h}$, $1A00_{h}$ or $1A01_{h}$) as o.
2. Revise the sub-index from 1 to 4 of PDO coordinated mapping parameter ($1600_{h, 1601\,h}$, $1A00_{h}$ or $1A01_{h}$).
3. Set the sub-index 0 of PDO coordinated mapping parameter(1600_{h} ,1601_{h},$1A00_{h}$ or $1A01_{h}$) as legal number(number of PDO's mapping objects)
4. PDO mapping completing.

There are multiple ways to transmit PDO:

- Synchronous（Synchronization by receiving SYNC object）

  Cycle: Transmit triggered after every 1 to 240 SYNC messages.

- Asynchronous

  Transmit triggered by special object event regulated in sub-object protocol.

Transmit type of PDO

Transmit Type	**Description**	**PDO**
0	Reserved	—
1~240	**SYNC**：It represents the number of SYNC objects between 2 PDOs.	TPDO/RPDO
240~253	Reserved	—
254	**Asynchronous**：If the content of PDO has changed, PDO transmit will be triggered.	TPDO
255	**Asynchronous**： The content of PDO will be periodically updated and transmitted.	TPDO/RPDO

One PDO could set a frozen time which is the shortest interval time between 2 continuous PDO. It could prevent the bus

from being occupied by amount of data with high priority. Frozen time is defined by 16 bit unsigned integer number and its unit is 100us.

One PDO could set a timing period. When the regulated time is violated, a PDO transmit could be triggered without a trigger bit. Object timing period is defined as 16 bit unsigned integer and its unit is 1ms.

PDO mapping case：

Map the 3 objects to PDO1 (transmit). PDO1 (transmit) is required to be asynchronous periodic type with period time as much as 10ms and frozen time as much as 2ms.

Object	Index — Sub-index	Description
statusword	6041_h – 00 h	Status word
modes_of_operation_display	6061_h – 00 h	Practical operational mode
Position_Acture_Value	6064_h – 00 h	Practical position

1）Clear number_of_mapped_objects

number_of_mapped_objects(1A00 h：00 h)= 0

2）Set the parameter for mapping objects

Index =6041 h　　Subin. = 00h Length = 10 h ⇒ 1st_mapped_object(1A00 h：01 h)= 60410010 h

Index =6061 h　　Subin. = 00h Length = 08 h ⇒ 2st_mapped_object(1A00 h：02 h)= 60610008 h

Index =60FD h　Subin. = 00h Length = 20 h ⇒ 3st_mapped_object(1A00 h：03 h) = 60FD0020 h

3）Set number_of_mapped_objects

number_of_mapped_objects(1A00 h：00 h)= 3

4）Set PDO communication parameter

PDO1（transmit）is asynchronous periodical type ⇒ transmission_type (1800 h：02 h)= FF h

Frozen time 2ms(20×100us) ⇒ inhibit_time (10A0 h：03 h)= 14 h

Period time:　10ms(10×1ms) ⇒ event_time (1800 h：05 h)= 0A h

5）PDO mapping complete.

PDO parameter

ProNet drive contains 4 transmit PDOs and 4 receive PDOs. The detailed communication parameter and mapping parameter of the first transmit/receive PDO is as below and those of the rest 3 transmit/receive PDO are the same as the first PDO.

Index	1800 h
Name	transmit_pdo_parameter_tpdo1
Object Code	RECORD
No. of Elements	4

Sub-Index	01 h
Description	cob_id_used_by_pdo_tpdo1
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	——
Value Range	181 h...1FF h, Bit 31 may be set
Default Value	181 h

Sub-Index	02 h
Description	transmission_type_tpdo1
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	——
Value Range	1...240,254,255
Default Value	255

Sub-Index	03 h
Description	inhibit_time_tpdo1
Data Type	UINT16
Access	RW
PDO Mapping	NO
Units	100μs
Value Range	——
Default Value	100

Sub-Index	05 h
Description	event_time_tpdo1
Data Type	UINT16
Access	RW
PDO Mapping	NO
Units	1ms
Value Range	——
Default Value	10

Index	1A00 h
Name	transmit_pdo_mapping_tpdo1
Object Code	RECORD
No. of Elements	2

Sub-Index	00 h
Description	number_of_mapped_objects_tpdo1
Data Type	UINT8
Access	RW
PDO Mapping	NO
Units	——
Value Range	0...4
Default Value	2

Sub-Index	01 h
Description	first_mapped_object_tpdo1
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	——
Value Range	——
Default Value	See table

Sub-Index	02 h
Description	second_mapped_object_tpdo1
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	——
Value Range	——

Default Value	See table

Sub-Index	03 h
Description	third_mapped_object_tpdo1
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	——
Value Range	——
Default Value	See table

Sub-Index	04 h
Description	fourth_mapped_object_tpdo1
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	——
Value Range	——
Default Value	See table

1、T-PDO1

Index	Comment	Type	Acc.	Default Value
1800 h _00 h	number of entries	UINT8	RO	04 h
1800 h _01 h	COB-ID used by PDO	UINT32	RW	00000181 h
1800 h _02 h	transmission type	UINT8	RW	FF h
1800 h _03 h	inhibit time (100 μs)	UINT16	RW	64 h
1800 h _05 h	event time (1ms)	UINT16	RW	0A h
1A00 h _00 h	number of mapped objects	UINT8	RW	02 h
1A00 h _01 h	first mapped object	UINT32	RW	60410010 h
1A00 h _02 h	second mapped object	UINT32	RW	60640020 h
1A00 h _03 h	third mapped object	UINT32	RW	00 h
1A00 h _04 h	fourth mapped object	UINT32	RW	00 h

2、T-PDO2

Index	Comment	Type	Acc.	Default Value
1801 h _00 h	number of entries	UINT8	RO	04 h
1801 h _01 h	COB-ID used by PDO	UINT32	RW	00000281 h
1801 h _02 h	transmission type	UINT8	RW	FF h
1801 h _03 h	inhibit time (100 μs)	UINT16	RW	64 h
1801 h _05 h	event time (1ms)	UINT16	RW	0A h
1A01 h _00 h	number of mapped objects	UINT8	RW	02 h
1A01 h _01 h	first mapped object	UINT32	RW	60640020 h
1A01 h _02 h	second mapped object	UINT32	RW	60610010 h
1A01 h _03 h	third mapped object	UINT32	RW	00 h
1A01 h _04 h	fourth mapped object	UINT32	RW	00 h

3、T-PDO3

Index	Comment	Type	Acc.	Default Value
1802 h _00 h	number of entries	UINT8	RO	04 h
1802 h _01 h	COB-ID used by PDO	UINT32	RW	00000281 h
1802 h _02 h	transmission type	UINT8	RW	FF h
1802 h _03 h	inhibit time (100 μs)	UINT16	RW	64 h
1802 h _05 h	event time (1ms)	UINT16	RW	0A h
1A02 h _00 h	number of mapped objects	UINT8	RW	02 h
1A02 h _01 h	first mapped object	UINT32	RW	60640020 h
1A02 h _02 h	second mapped object	UINT32	RW	60610010 h
1A02 h _03 h	third mapped object	UINT32	RW	00 h
1A02 h _04 h	fourth mapped object	UINT32	RW	00 h

4、T-PDO4

Index	Comment	Type	Acc.	Default Value
1803 h _00 h	number of entries	UINT8	RO	04 h
1803 h _01 h	COB-ID used by PDO	UINT32	RW	00000281 h
1803 h _02 h	transmission type	UINT8	RW	FF h
1803 h _03 h	inhibit time (100 μs)	UINT16	RW	64 h
1803 h _05 h	event time (1ms)	UINT16	RW	0A h
1A03 h _00 h	number of mapped objects	UINT8	RW	02 h
1A03 h _01 h	first mapped object	UINT32	RW	60640020 h
1A03 h _02 h	second mapped object	UINT32	RW	60610010 h
1A03 h _03 h	third mapped object	UINT32	RW	00 h
1A03 h _04 h	fourth mapped object	UINT32	RW	00 h

If transmit tye is 254 (if PDO content has changed,trigger will be sent by PDO),using the following object can shield parts of PDO changers.Only when the un-shield bit has changed，PDO is occur.If wants shielding any bit, the corresponding bit of object write to 0.

tpdo_1_transmit_mask

Index	Comment	Type	Acc.	Default Value
2000 h _00 h	number of entries	UINT8	RO	02 h
2000 h _01 h	tpdo_1_transmit_mask_low	UINT32	RW	FFFFFFFF h
2000 h _02 h	tpdo_1_transmit_mask_high	UINT32	RW	FFFFFFFF h

tpdo_2_transmit_mask

Index	Comment	Type	Acc.	Default Value
2001 h _00 h	number of entries	UINT8	RO	02 h
2001 h _01 h	tpdo_2_transmit_mask_low	UINT32	RW	FFFFFFFF h
2001 h _02 h	tpdo_2_transmit_mask_high	UINT32	RW	FFFFFFFF h

tpdo_3_transmit_mask

Index	Comment	Type	Acc.	Default Value
2002 h _00 h	number of entries	UINT8	RO	02 h
2002 h _01 h	tpdo_1_transmit_mask_low	UINT32	RW	FFFFFFFF h
2002 h _02 h	tpdo_1_transmit_mask_high	UINT32	RW	FFFFFFFF h

tpdo_4_transmit_mask

Index	Comment	Type	Acc.	Default Value
2003 h _00 h	number of entries	UINT8	RO	02 h
2003 h _01 h	tpdo_2_transmit_mask_low	UINT32	RW	FFFFFFFF h
2003 h _02 h	tpdo_2_transmit_mask_high	UINT32	RW	FFFFFFFF h

1、R-PDO1

Index	Comment	Type	Acc.	Default Value
1400 h _00 h	number of entries	UINT8	RO	02 h
1400 h _01 h	COB-ID used by PDO	UINT32	RW	00000201 h
1400 h _02 h	transmission type	UINT8	RW	FF h
1600 h _00 h	number of mapped objects	UINT8	RW	02 h
1600 h _01 h	first mapped object	UINT32	RW	60400010 h
1600 h _02 h	second mapped object	UINT32	RW	60FF0020 h
1600 h _03 h	third mapped object	UINT32	RW	00 h
1600 h _04 h	fourth mapped object	UINT32	RW	00 h

2、R-PDO2

Index	Comment	Type	Acc.	Default Value
1401 h _00 h	number of entries	UINT8	RO	02 h
1401 h _01 h	COB-ID used by PDO	UINT32	RW	00000301 h
1401 h _02 h	transmission type	UINT8	RW	FF h
1601 h _00 h	number of mapped objects	UINT8	RW	02 h
1601 h _01 h	first mapped object	UINT32	RW	60FF0020 h
1601 h _02 h	second mapped object	UINT32	RW	60600010 h
1601 h _03 h	third mapped object	UINT32	RW	00 h
1601 h _04 h	fourth mapped object	UINT32	RW	00 h

3、R-PDO3

Index	Comment	Type	Acc.	Default Value
1402 h _00 h	number of entries	UINT8	RO	02 h
1402 h _01 h	COB-ID used by PDO	UINT32	RW	00000301 h
1402 h _02 h	transmission type	UINT8	RW	FF h
1602 h _00 h	number of mapped objects	UINT8	RW	02 h
1602 h _01 h	first mapped object	UINT32	RW	60FF0020 h
1602 h _02 h	second mapped object	UINT32	RW	60600010 h
1602 h _03 h	third mapped object	UINT32	RW	00 h
1602 h _04 h	fourth mapped object	UINT32	RW	00 h

4、R-PDO4

Index	Comment	Type	Acc.	Default Value
1403 h _00 h	number of entries	UINT8	RO	02 h
1403 h _01 h	COB-ID used by PDO	UINT32	RW	00000301 h
1403 h _02 h	transmission type	UINT8	RW	FF h
1603 h _00 h	number of mapped objects	UINT8	RW	02 h
1603 h _01 h	first mapped object	UINT32	RW	60FF0020 h
1603 h _02 h	second mapped object	UINT32	RW	60600010 h
1603 h _03 h	third mapped object	UINT32	RW	00 h
1603 h _04 h	fourth mapped object	UINT32	RW	00 h

3.4 SYNC message

Synchronization object is used for controlling data synchronize transmit. For example: starting synchronously several axises. The transmition of synchronous message is based on Producer-Customer model. All the nodes of synchronous PDO can receive (at the same time) the message as customer and synchronize other node.

General mode:

CANopen suggests a COB-ID with highest priority to ensure that synchronized signal could be transmitted properly. Without transferring data, SYNC message could be as short as possible.

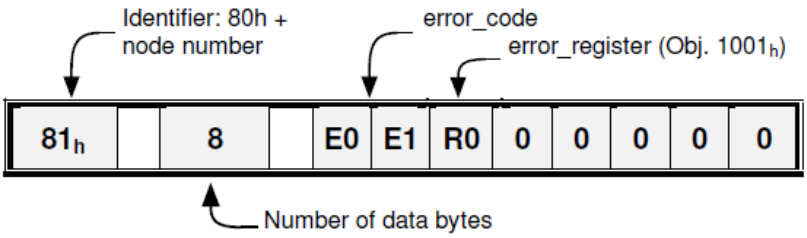
The identifier the servo controller receives SYNC messages are fixed to 080h. The identifier can be read via the object **cob_id_sync**.

Index	1005 $_{h}$
Name	cob_id_sync
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	NO
Units	——
Value Range	80000080 $_{h}$, 00000080 $_{h}$
Default Value	00000080 $_{h}$

3.5 Emergency message

When an alarm occurs to drive, CANopen will initiate an Emergency message to inform the current drive type and error code to clients. Error code displayed on panel can be read on low byte of 603Fh object.

The structure of Emergency message：



Alarm code

error_code（hex）	Description
2310	Over current
3100	Instantaneous power failure
3110	Over voltage
3120	Under voltage
5080	RAM exception
5210	AD sampling error
5420	Regenerative resistor error
5421	Regenerative resistor exception
5581	Parameter checksum exception
5582	electric gear error
5583	Motor type or drive type error
6100	Illegal error code
6120	PDO mapping error
6300	CAN communication error(Address or communication baud rate error)
7303	serial encoder error
7305	Incremental encoder error
7380	Resolver error
8100	CAN communication exception
8110	CAN bus overflow
8120	PASSIVE CAN bus turn to PASSIVE
8130	Heartbeat error
8140	CAN BUS OFF
8200	Length of CAN messages error
8210	Length of receiving PDO error
8311	Overload alarm
8480	Over speed alarm
8681	Forward run prohibited POT
8682	Reverse run prohibited NOT

Relevant parameter：

Index	1003 h
Name	pre_defined_error_field
Object Code	ARRAY
No. of Elements	4
Data Type	UINT32

Sub-Index	01 h
Description	standard_error_field_0
Access	RO
PDO Mapping	NO
Units	——
Value Range	——
Default Value	——

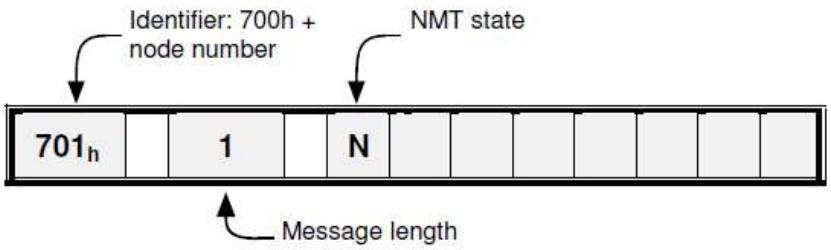
Sub-Index	02 h
Description	standard_error_field_1
Access	RO
PDO Mapping	NO
Units	——
Value Range	——
Default Value	——

Sub-Index	03 h
Description	standard_error_field_2
Access	RO
PDO Mapping	NO
Units	——
Value Range	——
Default Value	——

Sub-Index	04 h
Description	standard_error_field_3
Access	RO
PDO Mapping	NO
Units	——
Value Range	——
Default Value	——

3.6 HEARTBEAT message

Structure of the heartbeat message：

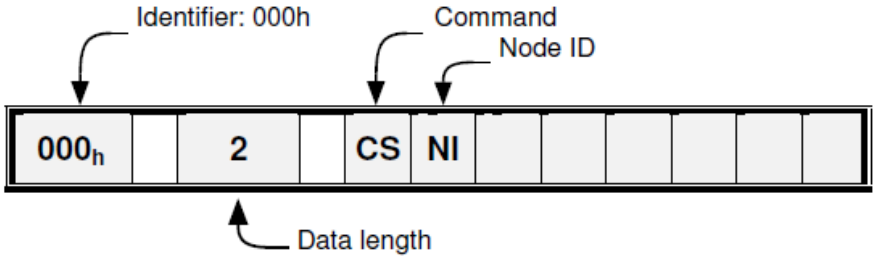


Relevant parameter：

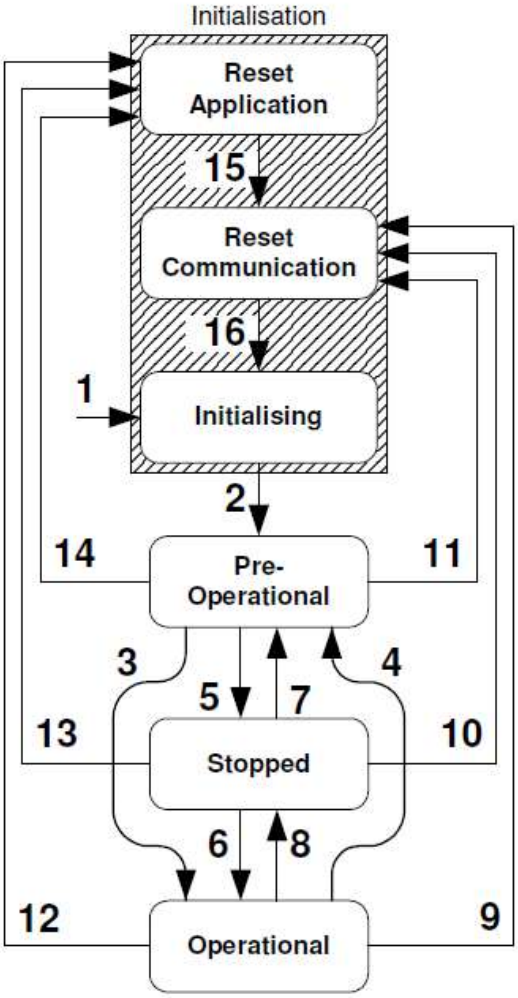
Index	1017 h
Name	producer_heartbeat_time
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	NO
Units	ms
Value Range	0 - 65535
Default Value	1000

3.7 Network management (NMT service)

Structure of the message：



NMT-State machine：

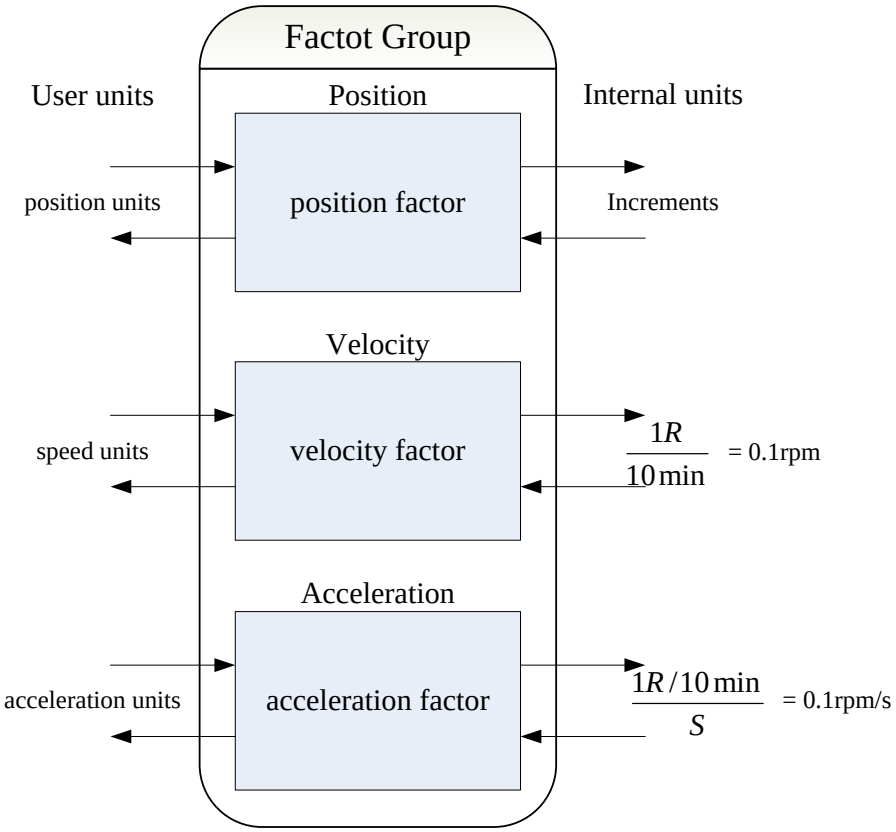


CS	Meaning	Transition	Target state
01_h	Start Remote Node	3, 6	**Operational**
02_h	Stop Remote Node	5, 8	**Stopped**
80_h	Enter Pre-Operational	4, 7	**Pre-Operational**
81_h	Reset Application	12, 13, 14	**Reset Application**
82_h	Reset Communication	9, 10, 11	**Reset Communication**

Name	Meaning	SDO	PDO	NMT
Reset Application	No communication. All CAN objects are set to their reset values (application parameter set).	-	-	-
Reset Communication	No communication. The CAN controller will be re-initialised.	-	-	-
Initialising	State after Hardware Reset. Reset of the CAN node, sending of the Bootup message	-	-	-
Pre-Operational	Communication via SDOs possible. PDOs inactive (No sending / receiving)	X	-	X
Operational	Communication via SDOs possible. PDOs active (sending / receiving)	X	X	X
Stopped	No communication except heartbeat + NMT	-	-	X

Chapter 4 Conversion factors (factor group)

Servo controllers will be used in a huge number of applications: As direct drive, with gear or for linear drives. To allow an easy parameterization for all kinds of applications, the servo controller can be parameterized in such a way that all values like the demand velocity refer to the driven side of the plant. The necessary calculation is done by the servo controller.



The default setting of the Factor Group is as follows:

Value	Name	Unit	Remark
Length	position units	Increments	Increments per revolution *
Velocity	speed units	1R /10min	0.1rpm
Acceleration	Acceleration units	1R/10min/s	0.1rpm/s
Jerk	jerk units	pulse/(s*100μs*100μs)	Range:1-20,more smaller,more smooth

***:** Common incremental encoder: 10000P/R

Resolver: 65536P/R

17 bit incremental encoder: 131072P/R

17 bit absolute encoder: 131072P/R

20 bit absolute encoder: 1048576P/R

4.1 Related parameters

Index	Object	Name	Type	Attr.
6093 h	ARRAY	position factor	UINT32	RW
6094 h	ARRAY	velocity factor	UINT32	RW
6097 h	ARRAY	acceleration factor	UINT32	RW

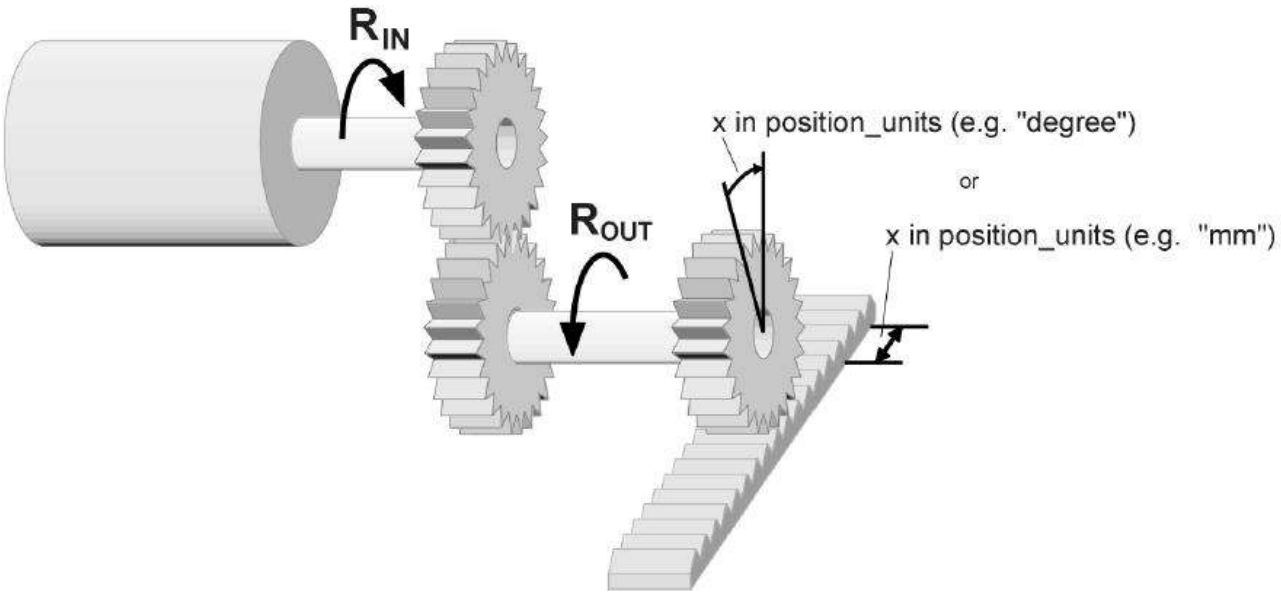
4.2 Position factor

The object **position factor** converts all values of length of the application from **Position units** into the internal unit **increments** (*encoder resolution* equals 1 Revolution). It consists of numerator and divisor:

Index	6093 h
Name	position factor
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01 h
Description	numerator
Access	RW
PDO Mapping	YES
Units	——
Value Range	——
Default Value	When power on, this value will be initiated to parameter Pn201

Sub-Index	02 h
Description	division
Access	RW
PDO Mapping	YES
Units	——
Value Range	——
Default Value	When power on, this value will be initiated to parameter Pn202



To calculate the **position factor** the following values are necessary:

gear_ratio Ratio between revolutions on the driving side (RIN) and revolutions on the driven side (ROUT).

feed_constant Ratio between revolutions on the driven side (ROUT) and equivalent motion in **position_units** (e.g. 1 rev = 360°)

The calculation of the **position_factor** is done with the following equation:

$$\textbf{position factor} = \frac{\text{numerator}}{\text{division}} = \frac{\text{gear\_ratio} * \text{encoder\_resolution}}{\text{feed\_constant}}$$

Note:

Encoder type	Encoder_resolution(Unit: Inc)
Common incremental encoder	10000
Resolver	65535
17 bit encoder	131072
20 bit encoder	1048576

4.3 Velocity factor

The object **velocity factor** converts all speed values of the application from **speed_units** into the internal unit **revolutions 0.1rpm**. It consists of numerator and divisor

Index	6094 $_{h}$
Name	velocity factor
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01 $_{h}$
Description	numerator
Access	RW
PDO Mapping	YES
Units	——
Value Range	——
Default Value	1

Sub-Index	02 $_{h}$
Description	division
Access	RW
PDO Mapping	YES
Units	——
Value Range	——
Default Value	1

In principle the calculation of the **velocity factor** is composed of two parts: A conversion factor from internal units of length into **position_units** and a conversion factor from internal time units into user defined time units (e.g. from seconds to minutes). The first part equals the calculation of the **position_factor.** For the second part another factor is necessary for the calculation:

time_factor_v Ratio between internal and user defined time units. (z.B. **1** ***min*** = **1/10** ***10 min***)

gear_ratio Ratio between revolutions on the driving side (RIN) and revolutions on the driven side (ROUT).

feed_constant Ratio between revolutions on the driven side (ROUT) and equivalent motion in position_units (e.g. 1 R = 360°)

The calculation of the **velocity factor** is done with the following equation:

$$\textbf{velocity factor} = \frac{\text{numerator}}{\text{division}} = \frac{\text{gear\_ratio} * \text{time\_factor\_v}}{\text{feed\_constant}}$$

4.4 Acceleration factor

The object **acceleration_factor** converts all acceleration values of the application from **acceleration_units** into the internal unit（0.1rpm）。It consists of numerator and divisor:

Index	6097 h
Name	acceleration factor
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01 h
Description	numerator
Access	RW
PDO Mapping	YES
Units	——
Value Range	——
Default Value	1

Sub-Index	02 h
Description	division
Access	RW
PDO Mapping	YES
Units	——
Value Range	——
Default Value	1

The calculation of the **acceleration_factor** is also composed of two parts: A conversion factor from internal units of length into **position_units** and a conversion factor from internal time units squared into user defined time units squared (e.g. from seconds2 to minutes2). The first part equals the calculation of the **position_factor.** For the second part another factor is necessary for the calculation

time_factor_a Ratio between internal time units squared and user defined time units squared

（z.B.： 1**min**^{2} = 1min*min = 60s*1min =60/10 **10min/s**）

gear_ratio Ratio between revolutions on the driving side (RIN) and revolutions on the driven side (ROUT).

feed_constant Ratio between revolutions on the driven side (ROUT) and equivalent motion in position_units (e.g. 1 R = 360°)

The calculation of the **acceleration_factor** is done with the following equation:

$$\textbf{acceleration factor} = \frac{\text{numerator}}{\text{division}} = \frac{\text{gear\_ratio} * \text{time\_factor\_a}}{\text{feed\_constant}}$$

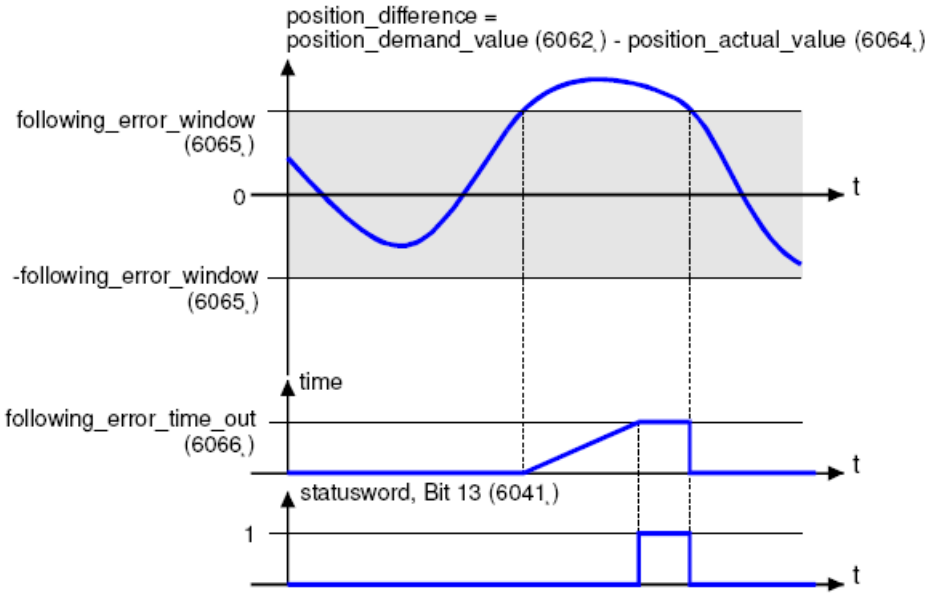
Chapter 5 Position control function

This chapter describes all parameters which are required for the position controller. The desired position value (**position_demand_value**) of the trajectory generator is the input of the position controller. Besides this the actual position value (**position_actual_value**) is supplied by the angle encoder (resolver, incremental encoder, etc.). The behaviour of the position controller can be influenced by parameters.

It is possible to limit the output quantity (**control_effort**) in order to keep the position control system stable. The output quantity is supplied to the speed controller as desired speed value. In the **Factor Group** all input and output quantities are converted from the application-specific units to the respective internal units of the controller

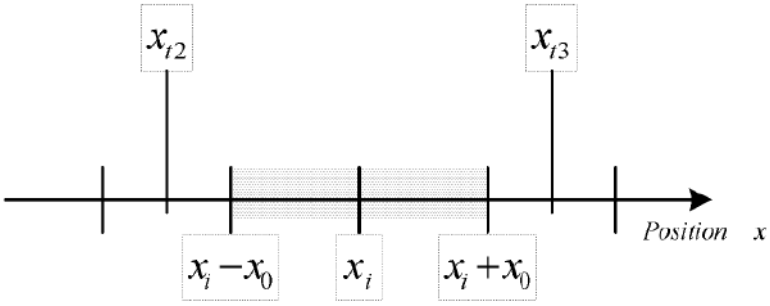
The following subfunctions are defined in this chapter:

1. Trailing error (Following Error)



Trailing error (Following Error) – Function Survey

The deviation of the actual position value (**position_actual_value**) from the desired position value (**position_demand_value)** is named trailing error. If for a certain period of time this trailing error is bigger than specified in the trailing error window (**following_error_window**) bit 13 (**following_error**) of the object **statusword** will be set to 1.



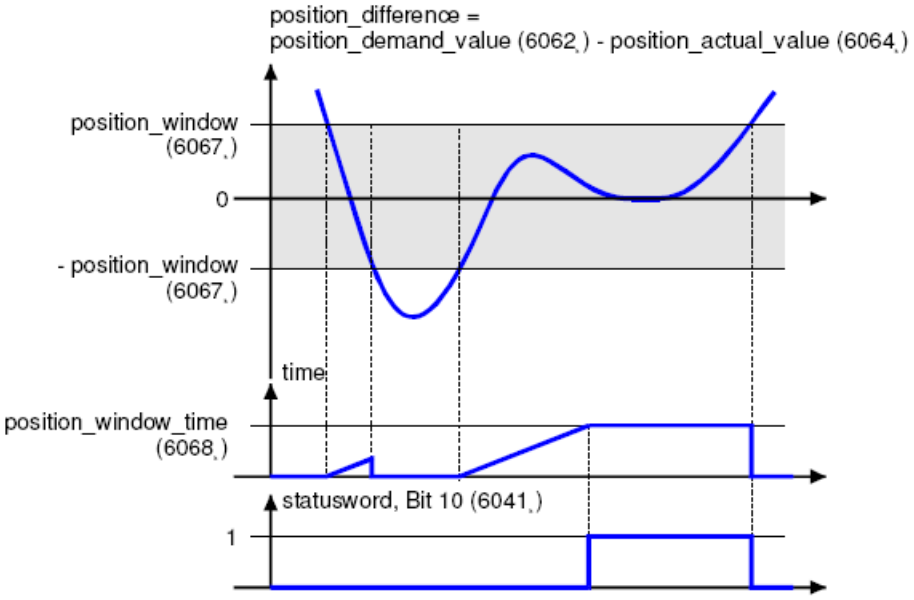
Trailing error (following error)

The permissible time can be defined via the object **following_error_time_out**.。 Figure above shows how the window function is defined for the message "following error".The range between xi-x0 and xi+x0 is defined symmetrically

around the desired position (**position_demand_value**) xi. For example the positions xt2 and xt3 are outside this window (**following_error_window**). If the drive leaves this window and does not return to the window within the time defined in the object **following_error_time_out** then bit 13 (**following_error**) in the **statusword** will be set to 1.

2**.** Position Reached

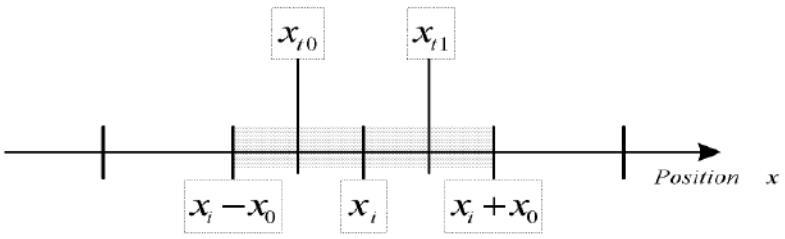
This function offers the chance to define a position window around the target position (**target_position**). If the actual position of the drive is within this range for a certain period of time – the **position_window_time –** bit 10 (**target_reached**) will be set to 1 in the statusword.



.

Position reached-function description

Figure below shows how the window function is defined for the message "position reached". The position range between xi-x0 and xi+x0 is defined symmetrically around the target position (**target_position**) xi. For example the positions xt0 and xt1 are inside this position window (**position_window**). If the drive is within this window a timer is started. If this timer reaches the time defined in the object **position_window_time** and the drive uninterruptedly was within the valid range between xi-x0 and xi+x0, bit 10 (**target_reached**) will be set in the **statusword**. As far as the drive leaves the permissible range, bit 10 is cleared and the timer is set to zero.



.

Position reached

Parameters:

Index	Object	Name	Type	Attr.
6062 h	VAR	position_demand_value	INT32	RO
6063 h	VAR	position_actual_value*	INT32	RO
6064 h	VAR	position_actual_value	INT32	RO
6065 h	VAR	following_error_window	UINT32	RW
6066 h	VAR	following_error_time_out	UINT16	RW
6067 h	VAR	position_window	UINT32	RW
6068 h	VAR	position_time	UINT16	RW
60FA h	VAR	control_effort	INT32	RO

Index	6062 h
Name	position_demand_value
Object Code	VAR
Data Type	INT32
Access	RO
PDO Mapping	YES
Units	position units
Value Range	--
Default Value	--

Index	6064 h
Name	position_ actual _value
Object Code	VAR
Data Type	INT32
Access	RO
PDO Mapping	YES
Units	position units
Value Range	--
Default Value	--

Index	6065 h
Name	following_error_window
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	position units
Value Range	0 – 7FFFFFFF h
Default Value	256

Index	6066 h
Name	following_error_time_out
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	YES
Units	Ms
Value Range	0 – 65535
Default Value	0

Index	60FA h
Name	control_effort
Object Code	VAR
Data Type	INT32
Access	RO
PDO Mapping	YES
Units	speed units
Value Range	--
Default Value	--

Index	6067 h
Name	position_window
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	position units
Value Range	--
Default Value	400

Index	6068 h
Name	position_time
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	YES
Units	Ms
Value Range	0 – 65535
Default Value	0

Chapter 6 Device control

The following chapter describes how to control the servo controller using CANopen, i.e. how to switch on the power stage or to reset an error.

6.1 State diagram (State machine)

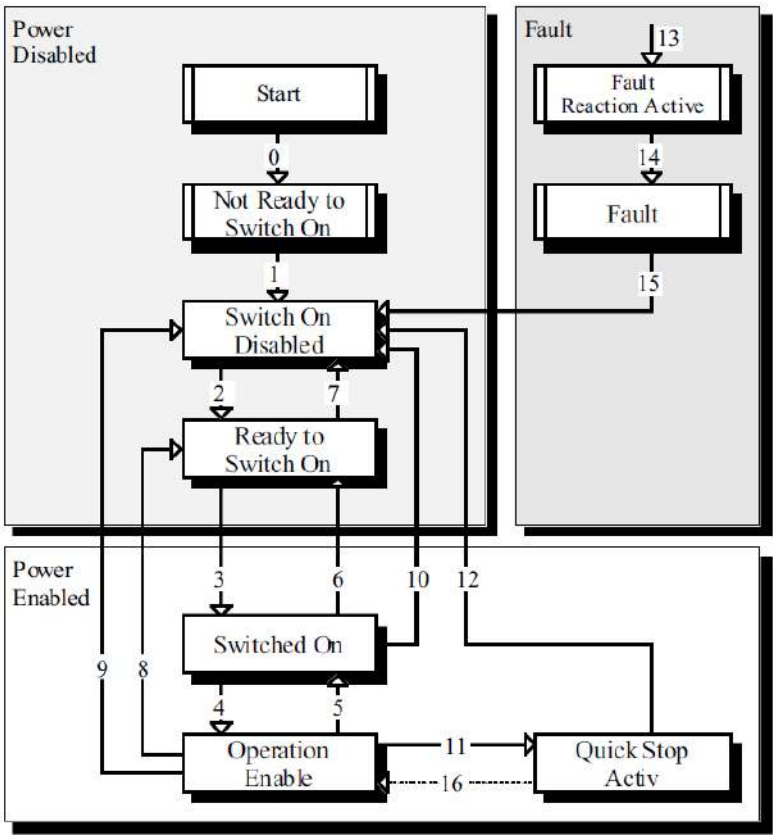
Using CANopen the complete control of the servo is done by two objects. Via the **controlword** the host is able to control the servo, as the status of the servo can be read out of the **statusword**. The following items will be used in this chapter:

State： The servo controller is in different states dependent on for instance if the power stage is alive or if an error has occurred. States defined under CANopen will be explained in this chapter.
Example: **SWITCH_ON_DISABLED**

State Transition： Just as the states it is defined as well how to move from one state to another (e.g. to reset an error). These state transitions will be either executed by the host by setting bits in the **controlword** or by the servo controller itself, if an error occurs for instance.

Command： To initiate a state transition defined bit combinations have to be set in the **controlword**. Such bit combination are called command. Example: **Enable Operation**

State diagram： All the states and all state transitions together form the so called state diagram: A survey of all states and the possible transitions between two states.。



State diagram of the servo controller

The state diagram can be divided into three main parts: "Power Disabled" means the power stage is switched off and "Power Enabled" the power stage is active. The area "Fault" contains all states necessary to handle errors of the controller. The most important states have been highlighted in the Figure: After switching on the servo controller initializes itself and reaches the state **SWITCH_ON_DISABLED** after all. In this state CAN communication is possible and the servo controller can be parameterized (e.g. the mode of operation can be set to "velocity control"). The power stage remains switched off and the motor shaft is freely rotatable. Through the state transitions 2, 3 and 4 – principally like the controller enable under CANopen - the state **OPERATION_ENABLE** will be reached. In this state the power stage is live and the servo controller controls the motor according to the parameterized mode of operation. Therefore previously ensure that the servo controller has been parameterized correctly and the according demand value is zero. The state transition 9 complies with disabling the power stage, i.e. the motor is freely rotatable.

Status	Description
Not Ready to Switch On	The servo controller executes its self-test. The CAN communication is not working
Switch On Disabled	The self-test has been completed. The CAN communication is activated..
Ready to Switch On	Servo driver is waiting for the state of Switch and servo motor is not at power stage
Switched On	The power stage is alive.
Operation Enable	The motor is under voltage and is controlled according to operational mode
Quick Stop Active	Servo driver will be stopped through its fixed way,
Fault Reaction Active	Servo driver tests error and will be stopped through its fixed way, with motor's power stage alive
Fault	An error has occurred. The power stage has been switched off.

6.2 Relevant parameters of device control

Index	Object	Name	Type	Attr.
6040 $_h$	VAR	controlword	UINT16	RW
6041 $_h$	VAR	statusword	UINT16	RO
605A $_h$	VAR	quick_stop_option_code	INT16	RW
605B $_h$	VAR	shutdown_option_code	INT16	RW
605C $_h$	VAR	disabled_operation_option_code	INT16	RW
605D $_h$	VAR	halt_option_code	INT16	RW
605E $_h$	VAR	fault_reaction_option_code	INT16	RW

6.2.1 Controlword

Index	6040 $_h$
Name	controlword
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	YES

Units	--
Value Range	--
Default Value	0

Controlword bit description is as below…

15 11	10 9	8	7	6 4	3	2	1	0
manufacturer specific	reserved	halt	Fault reset	Operation mode specific	Enable operation	Quick stop	Enable voltage	Switch on

Bit0 ~ 3 and Bit7：

Transmit of status machine is triggered by 5 bits coordinated control code as below…

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

Device control list

Note: X means this bit could be ignored.

Bit4、5、6、8：

The definition of this 4 bit is different in different control mode…

Bit	Control Mode		
	profile position mode	profile velocity mode	homing mode
4	new_set_point	reserved	start_homeing_operation
5	change_set_immediatly	reserved	reserved
6	abs/rel	reserved	reserved
8	Halt	Halt	Halt

Other bits: all reserved

6.2.2 Statusword

Index	6041 h
Name	statusword
Object Code	VAR
Data Type	UINT16
Access	RO
PDO Mapping	YES

Units	--
Value Range	--
Default Value	--

Explanation of statusword bit is as below:

bit	name
0	Ready to switch on
1	Switched on
2	Operation enabled
3	Fault
4	Voltage enabled
5	Quick stop
6	Switch on disabled
7	Warning
8	Not used now
9	Remote
10	Target reached
11	Internal limit active
13~12	Operation mode specific
15~14	Not used now

Bit0 ~ 3 , Bit5 and Bit6：

The combination of this bit indicates the status of drives.

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

Bit4： Voltage enabled

Main power is on when this bit is 1.

Bit5： Quick stop

Driver will follow setting （605A $_h$：quick_stop_option_code）to halt when this bit is 0.

Bit7： Warning

Driver detects alarm when this bit is 1.

Bit9： Warning

Servo can deal with Controlword when this bit is 1 and CANOPEN is enabled.

Bit10： Target reached

In different control modes the meaning of this bit is different.

In profile position mode, when set position is reached, this bit is set. When Halt is booted, speed is reduced to 0 and this bit will be set. When new position is set, this bit will be cleared.

In profile Velocity Mode, when the speed reaches the targeted speed, this bit will be set. When Halt is booted and speed is reduced to 0, this bit is set.

Bit11： Internal limit active

When this bit is 1, it indicates that internal torque has surpassed the set value,or reached the max.forward/reverse run.It can be confirmed by reading object 60FDh（digital inputs）.

Bit12, 13：

These 2 bits mean different in different control mode…

Bit	Control mode		
	profile position mode	profile velocity mode	homing mode
12	Set-point acknowledge	Speed	Homing attained
13	Following error	Max slippage error	Homing error

Other bits：

All reserved.

6.2.3 Shutdown_option_code

The object **shutdown_option_code** determines the behaviour if the state transition 8 (from **OPERATION ENABLE** to **READY TO SWITCH ON)** will be executed.

Index	605B $_{h}$
Name	shutdown_option_code
Object Code	VAR
Data Type	INT16
Access	RW
PDO Mapping	NO
Units	--
Value Range	0,1
Default Value	0

Value	Name
0	Power stage will be switched off. Motor is freely rotatable.
1	Switch off the power stage after the motor stops deceleration.

6.2.4 Disable_operation_option_code

The object **disable_operation_option_code** determines the behaviour if the state transition 5 (from **OPERATION ENABLE** to **SWITCHED ON)** will be executed.

Index	605C $_{h}$
Name	disable_operation_option_code
Object Code	VAR
Data Type	INT16
Access	RW
PDO Mapping	NO
Units	--
Value Range	0,1
Default Value	0

Value	Description
0	Power stage will be switched off. Motor is freely rotatable.。
1	Switch off the power stage after the motor stops deceleration.

6.2.5 Quick_stop_option_code

The object **quick_stop_option_code** determines the behaviour if a **Quick Stop** will be executed.

Index	605A $_{h}$
Name	quick_stop_option_code
Object Code	VAR
Data Type	INT16
Access	RW
PDO Mapping	NO
Units	--
Value Range	0,1,2,5,6
Default Value	0

value	Description
0	Power stage will be switched off. Motor is freely rotatable.
1	Switch off the power stage after the motor stops deceleration.
2	Power stage will be shut down after the motor decelerates to still urgently.
5	QuickStop is alive after the motor decelerates to still.
6	QuickStop is alive after the motor decelerates urgently to still.

6.2.6 Halt_option_code

Halt_option_code determines how to stop when bit.8 (halt) of controlword is set to 1.

Index	605D $_h$
Name	halt_option_code
Object Code	VAR
Data Type	INT16
Access	RW
PDO Mapping	NO
Units	--
Value Range	1,2
Default Value	0

Value	Description
1	The motor decelerates to still.
2	The motor decelerates urgently to still

6.2.7 Fault_reaction_option_code

When an error is occurred, fault_reation_option_code determines how to stop.

Index	605E $_h$
Name	fault_reaction_option_code
Object Code	VAR
Data Type	INT16
Access	RW
PDO Mapping	NO
Units	--
Value Range	0
Default Value	0

Value	Description
0	Power stage will be switched off. Motor is freely rotatable.

Chapter 7 Control mode

ProNet/EDS/ETS currently supports 5 control modes in CANopen DSP402：

HOMING MODE

PROFILE VELOCITY MODE

PROFILE TORQUE MODE

PROFILE POSITION MODE

INTERPOLATED POSITION MODE

This chapter mainly describes four control modes as above.

7.1 Relevant parameter of control mode

Index	Object	Name	Type	Attr.
6060 h	VAR	modes_of_operation	INT8	RW
6061 h	VAR	modes_of_operation_display	INT8	RO

7.1.1 Modes_of_operation

Drive control mode will be determined by parameters in modes_of_operation.

Index	6060 h
Name	modes_of_operation
Object Code	VAR
Data Type	INT8
Access	RW
PDO Mapping	YES
Units	--
Value Range	1,3,4,6,7
Default Value	1

Value	Description
1	PROFILE POSITION MODE
3	PROFILE VELOCITY MODE
4	PROFILE TORQUE MODE
6	HOMING MODE
7	INTERPOLATION MODE

7.1.2 Modes_of_operation_display

Drive current control mode could be read from parameters in modes_of_operation_display.

Index	6061 h
Name	modes_of_operation_display
Object Code	VAR
Data Type	INT8
Access	RO
PDO Mapping	YES
Units	--
Value Range	1,3,4,6,7
Default Value	1

Note：The current control mode could be only known from parameters in modes_of_operation_display

7.2 Homing mode

PRONET servo drive currently supports multiple homing mode and users could choose the suitable homing mode. For example, if an incremental encoder is applied in servomotor, then homing mode of Zero impulse could be chosen and if serial encoder or resolver is applied in servomotor then Zero impulse homing mode couldn't be selected.

The user can determine the velocity, acceleration, and the kind of homing operation. After the servo controller has found its reference the zero position can be moved to the desired point via the object home_offset （607C h）.

7.2.1 Control word of homing mode

15 ~ 9	8	7 ~ 5	4	3 ~ 0
*	Halt	*	home_operation_start	*

*: referred to previous chapters …

Name	Value	Description
Homing operation start	0	Homing mode inactive
	0 → 1	Start homing mode
	1	Homing mode active
	1 → 0	Interrupt homing mode
Halt	0	Execute the instruction of bit 4
	1	Stop axle with homing acceleration

7.2.2 Status word of homing mode

15 ~ 14	13	12	11	10	9 ~ 0

*	homing_error	homing_attained	*	target_reached	*

*: referred to previous chapters

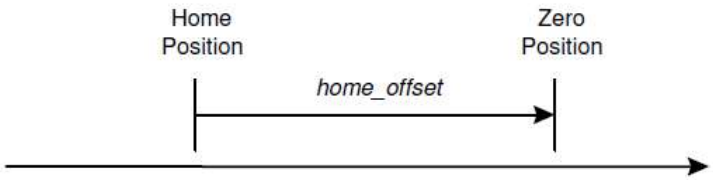
Name	Value	Description
Target reached	0	Halt = 0: Home position not reached Halt = 1: Axle decelerates
	1	Halt = 0: Home position reached Halt = 1: Axle has velocity 0
Homing attained	0	Homing mode not yet completed
	1	Homing mode carried out successfully
Homing error	0	No homing error
	1	Homing error occurred; Homing mode carried out not successfully; The error cause is found by reading the error code

7.2.3 Relevant parameter of homing mode

Index	Object	Name	Type	Attr.
607C h	VAR	home_offset	INT32	RW
6098 h	VAR	homing_method	INT8	RW
6099 h	ARRAY	homing_speeds	UINT32	RW
609A h	VAR	homing_acceleration	INT32	RW

home_offset

The object **home_offset** determines the displacement of the zero position to the limit resp. reference switch position.



Index	6098 h
Name	homing_method
Object Code	VAR
Data Type	INT8
Access	RW
PDO Mapping	YES
Units	--
Value Range	1-4，7-14*，17-20，23-30*，35
Default Value	1

Note：

*：Only some servodrives support homing_method.

homing_method

The negative and positive limit switch, the reference switch and the (periodic) zero impulse of the angle encoder.

Index	6098 $_{h}$
Name	homing_method
Object Code	VAR
Data Type	INT8
Access	RW
PDO Mapping	YES
Units	--
Value Range	1-14*，17-22，23-30*，33-35
Default Value	1

Note:* means only some servo models support this home mode.

Homing method value description

Value	Direction	Target	Reference point for Home position	DS402
1	Negative	NOT	Zero impulse	1
2	Positive	POT	Zero impulse	2
3	Negative	Reference switch	Zero impulse	3
4	Positive	Reference switch	Zero impulse	4
5	Negative	Reference switch	Zero impulse	5
6	Positive	Reference switch	Zero impulse	6
7	Positive	Reference switch	Zero impulse	7
8	Positive	Reference switch	Zero impulse	8
9	Positive	Reference switch	Zero impulse	9
10	Positive	Reference switch	Zero impulse	10
11	Negative	Reference switch	Zero impulse	11
12	Negative	Reference switch	Zero impulse	12
13	Negative	Reference switch	Zero impulse	13
14	Negative	Reference switch	Zero impulse	14
17	Negative	NOT	NOT	17
18	Positive	POT	POT	18
19	Negative	Reference switch	Reference switch	19
20	Positive	Reference switch	Reference switch	20
21	Negative	Reference switch	Reference switch	21
22	Positive	Reference switch	Reference switch	22
23	Positive	Reference switch	Reference switch	23
24	Positive	Reference switch	Reference switch	24
25	Positive	Reference switch	Reference switch	25
26	Positive	Reference switch	Reference switch	26
27	Negative	Reference switch	Reference switch	27
28	Negative	Reference switch	Reference switch	28
29	Negative	Reference switch	Reference switch	29
30	Negative	Reference switch	Reference switch	30
33	Negative	Current position	Zero impulse	33

34	Positive	Current position	Zero impulse	34
35	--	Current position	Current position	35
-4	Positive	Target torque	Zero impulse	Reserved
-3	Negative	Target torque	Zero impulse	Reserved
-2	Positive	Target torque	Target torque	Reserved
-1	Negative	Target torque	Target torque	Reserved

homing_speeds

There are two kinds of speeds required to find reference point, speed during search for switch and speed during search for zero.

Index	6099 $_{h}$
Name	homing_speeds
Object Code	ARRAY
No. of Elements	2
Data Type	INT32

Sub-Index	01 $_{h}$
Name	speed_during_search_for_switch
Object Code	VAR
Data Type	INT32
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	--
Default Value	0

Sub-Index	02 $_{h}$
Name	speed_during_search_for_zero
Object Code	VAR
Data Type	INT32
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	--
Default Value	100

Pn207（stopper torque）

It is used for homing method -4、-3、-2、-1.When the drive hits an end so that the torque set in Pn207 is reached for the blocking time set in Pn208,movement in the opposite direction or makes the current position for the origin.

Index	3049_h
Name	Pn207（stopper torque）
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	NO
Units	1% rated torque
Value Range	0-200
Default Value	20

Pn208（blocking time）

It is used for homing method -4、-3、-2、-1.When the drive hits an end so that the torque set in Pn207 is reached for the blocking time set in Pn208,movement in the opposite direction or makes the current position for the origin.

Index	$304A_h$
Name	Pn208（Blocking time）
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	NO
Units	0.125ms
Value Range	0-10000
Default Value	100

homing_acceleration

The objects **homing_acceleration** determine the acceleration which is used for all acceleration and deceleration operations during the search for reference.

Index	609A **h**
Name	homing_acceleration
Object Code	VAR
Data Type	INT32
Access	RW
PDO Mapping	YES
Units	acceleration units
Value Range	--
Default Value	100000

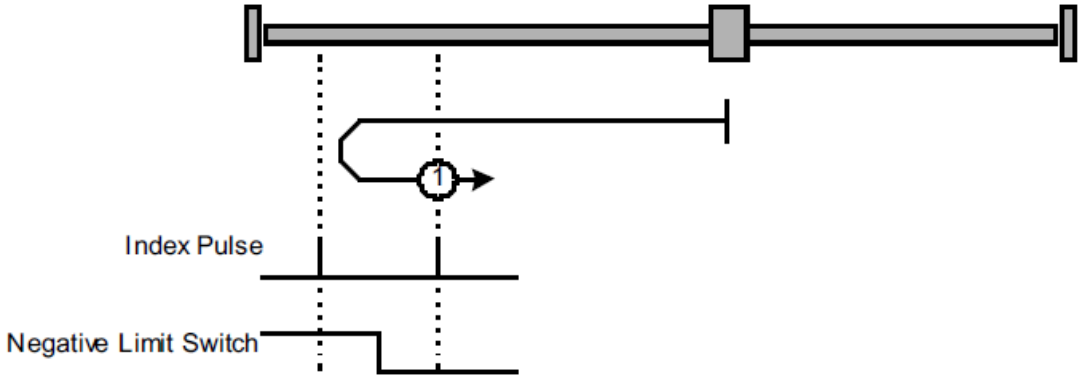
7.2.4 Homing sequences

Method 1: Using egative limit switch and zero impulse evaluation

A：When homing mode is enabled, If negative limit switch N-OT=0, the drive first moves relatively quick into the negative direction until it reaches the negative limit switch. This is displayed in the diagram by the rising edge. Afterwards the drive slowly returns, and stops until reaches the falling edge.

B: When homing mode is enabled, If negative limit switch N-OT=1, the drive first moves slowly into the positve direction

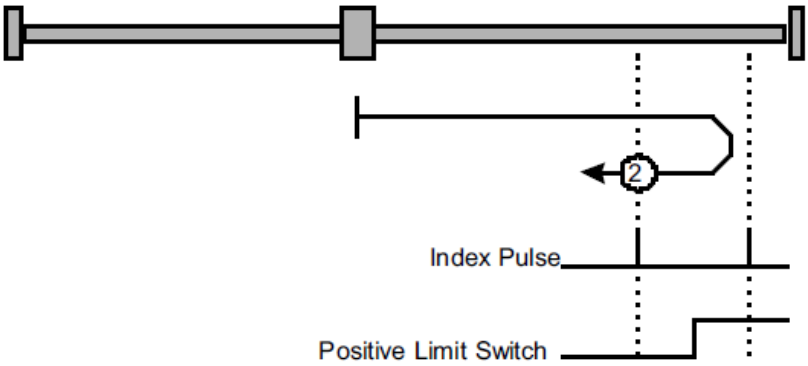
until reaches the falling edge.



Method 2: Using positive limit switch and zero impulse evaluation

A: When homing mode is enabled, If positive limit switch P-OT=0, the drive first moves relatively quick into the positive direction until it reaches the positive limit switch. This is displayed in the diagram by the rising edge. Afterwards the drive slowly returns, and stops until reaches the falling edge.

B: When homing mode is enabled, If positive limit switch P-OT=1, the drive first moves slowly into the negative direction until reaches the falling edge.



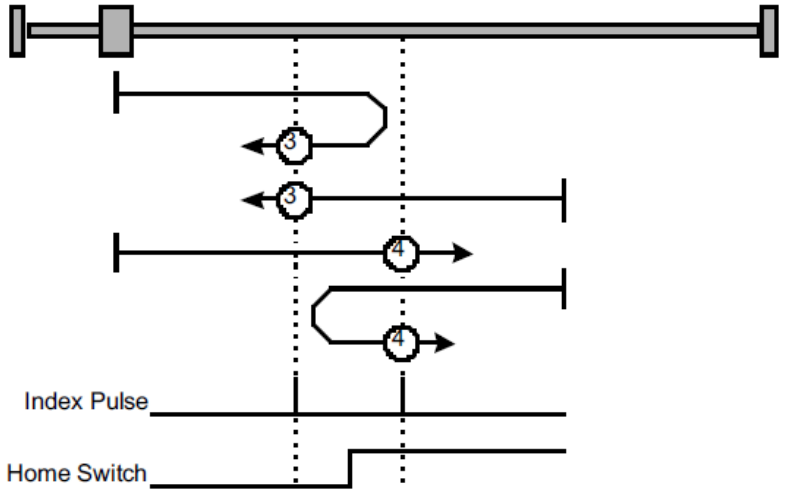
Methods 3 and 4: Using positive reference switch and zero impulse evaluation

Methods 3 A: When homing mode is enabled, If positive reference switch H-S=0, the drive first moves relatively quick into the positive direction until it reaches the positive reference switch. This is displayed in the diagram by the rising edge. Afterwards the drive slowly returns, and stops until reaches the falling edge.

Methods 3 B: When homing mode is enabled, If positive reference switch H-S =1, the drive first moves slowly into the negative direction until reaches the falling edge.

Methods 4 A: When homing mode is enabled, If positive reference switch H-S =0, the drive first moves slowly into the positive direction until reaches the rising edge.

Methods 4 B: When homing mode is enabled, If positive reference switch H-S=1, the drive first moves relatively quick into the negative direction until it reaches the positive reference switch. This is displayed in the diagram by the falling edge. Afterwards the drive slowly returns, and stops until reaches the rising edge.



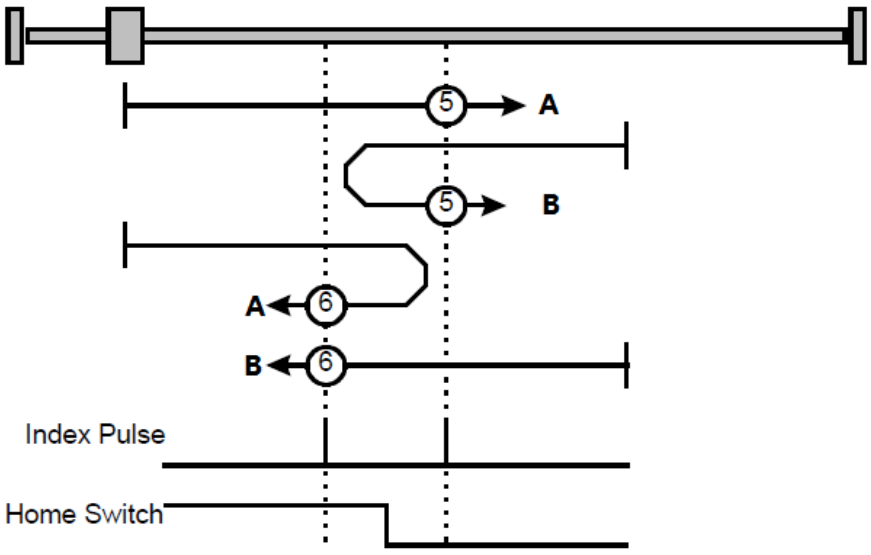
Methods 5 and 6: Using negative reference switch and zero impulse evaluation

Methods 5 A: When homing mode is enabled, If negative reference switch H-S =1, the drive first moves slowly into the positive direction until reaches the zero impulse evaluation. This is displayed in the diagram by the falling edge of H-S.

Methods 5 B: When homing mode is enabled, If negative reference switch H-S=0, the drive first moves relatively quick into the negative direction until it reaches the negative reference switch This is displayed in the diagram by the rising edge. Afterwards the drive slowly returns, and stops until reaches the zero impulse evaluation. This is displayed in the diagram by the falling edge of H-S.

Methods 6 A: When homing mode is enabled, If negative reference switch H-S=1, the drive first moves relatively quick into the positive direction until it reaches the negative reference switch. This is displayed in the diagram by the falling edge. Afterwards the drive slowly returns, and stops until reaches the zero impulse evaluation. This is displayed in the diagram by the rising edge of H-S.

Methods 6 B: When homing mode is enabled, If negative reference switch H-S =0, the drive first moves slowly into the positive direction, and stops until reaches the zero impulse evaluation. This is displayed in the diagram by the rising edge of H-S.



Methods 7 ~ 14 Using reference switch , limit switch and zero impulse evaluation

Methods 7~14 use the reference switch which is only active over parts of the distance.

（1） Use positive limit switch P-OT

If this method 7~10 is used the drive first moves relatively quick into the positive direction

Methods 7A：When homing mode is enabled, If reference switch H-S=0, the drive first moves relatively quick into the positive direction ,not reaches positive limit switch ,until it reaches the reference switch H-S. This is displayed in the diagram by the rising edge. Afterwards the drive slowly returns, and stops until reaches the falling edge.

Methods 7B: When homing mode is enabled, If reference switch H-S =1, the drive first moves slowly into the negative direction until reaches the falling edge.

Methods 7C：When homing mode is enabled, If reference switch H-S=0, the drive first moves relatively quick into the positive direction , and reaches positive limit switch .The drive moves quickly into the negative direction.When reachinig the rising edge of H-S ,the drive moves slowly , and moves into the negative direction until reaches the falling edge of H-S.

Methods 8 A：When homing mode is enabled, If reference switch H-S=0, the drive first moves relatively quick into the positive direction ,not reaches positive limit switch , Afterwards the drive moves slowly into positive derection when reaches the rising edge of H-S, and stops until reaches the zero impulse evaluation.

Methods 8 B: When homing mode is enabled, If reference switch H-S =1, the drive first moves slowly into the negative direction until reaches the falling edge of H-S. Then moves slowly into the positive direction, stops until reaches the zero impulse evaluation. This is displayed in the diagram by the H-S rising edge.

Methods 8 C：When homing mode is enabled, If reference switch H-S=0, the drive first moves relatively quick into the positive direction ,reaches positive limit switch ,Afterwards the drive moves quickly into the negative derection,until reaches the rising edge of H-S. The drive slows down, and moves into into the negative derection.Reaches the falling edge of H-S,the drive returns into positive derection,until reaches the zero impulse evaluation.This is displayed in the diagram by the H-S rising edge.

Methods 9 A：When homing mode is enabled, If reference switch H-S=0, the drive first moves relatively quick into the positive direction ,not reaches positive limit switch. Afterwards the drive moves slowly into positive derection when reaches the rising edge of H-S. The drive slows down to stop until reaches the falling edge of H-S.Then drive returns slowly,and stops until reaches the zero impulse evaluation.This is displayed in the diagram by the H-S rising edge.

Methods 9 B: When homing mode is enabled, If reference switch H-S =1, the drive first moves slowly into the postive direction until reaches the falling edge of H-S. Then moves slowly into the negative direction, stops until reaches the zero impulse evaluation. This is displayed in the diagram by the H-S rising edge.

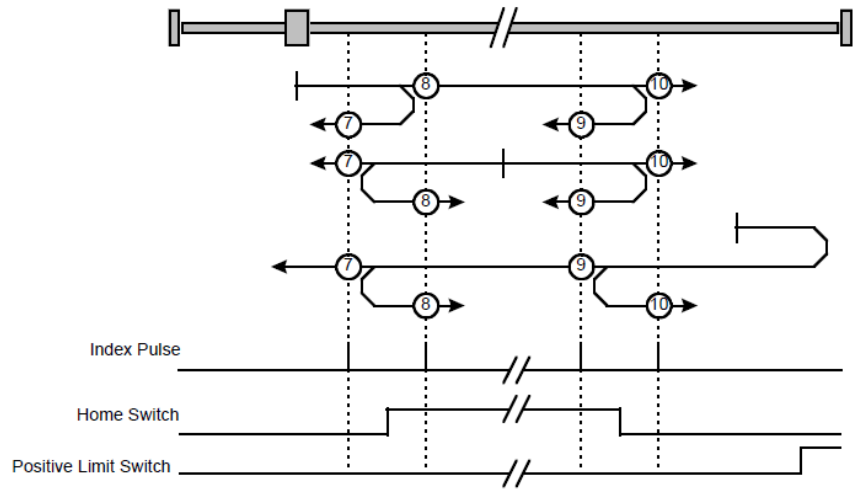
Methods 9 C：When homing mode is enabled, If reference switch H-S=0, the drive first moves relatively quick into the positive direction ,reaches positive limit switch ,Afterwards the drive moves quickly into the negative derection,until reaches the rising edge of H-S. The drive slows down, and moves into into the negative derection, and stops until reaches the zero impulse evaluation.

Methods 10 A：When homing mode is enabled, If reference switch H-S=0, the drive first moves relatively quick into the positive direction, not reaches positive limit switch. Afterwards the drive moves slowly into positive derection when reaches the rising edge of H-S.If reaches the falling edge of H-S,the drive moves slowly into positive derection until reaches the zero impulse evaluation.

Methods 10 B: When homing mode is enabled, If reference switch H-S =1, the drive first moves slowly into the positive direction until reaches the zero impulse evaluation. This is displayed in the diagram by the H-S falling edge.

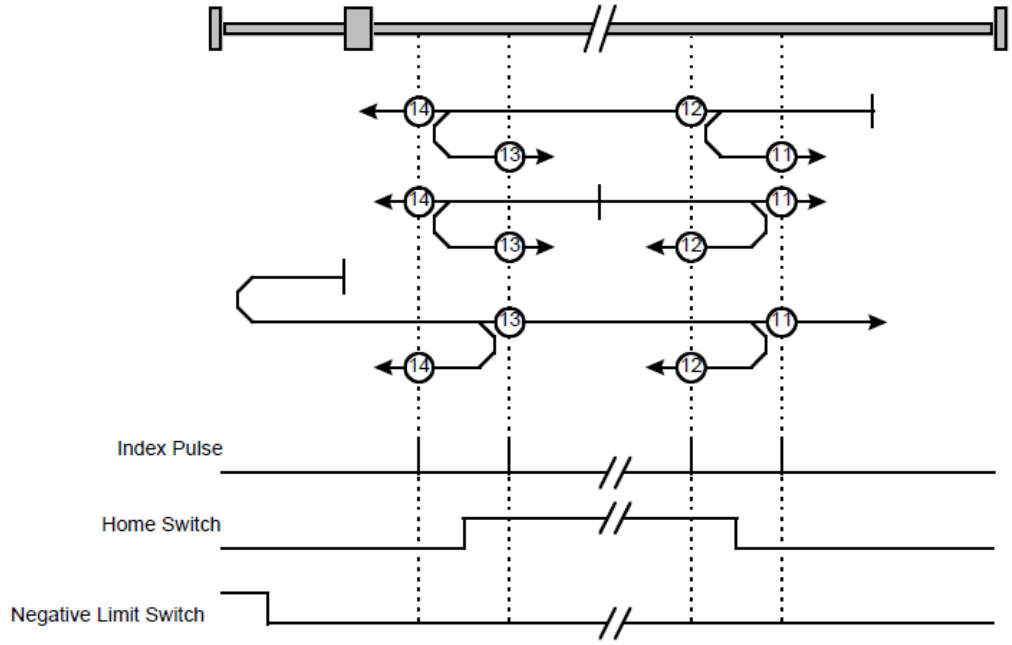
Methods 10 C：When homing mode is enabled, If reference switch H-S=0, the drive first moves relatively quick into the positive direction ,reaches positive limit switch ,Afterwards the drive moves quickly into the negative derection,until reaches the rising edge of H-S. The drive slows down to stop.Then the drive returns slowly to the positive derection, stops

until reaches the zero impulse evaluation. This is displayed in the diagram by the H-S falling edge.



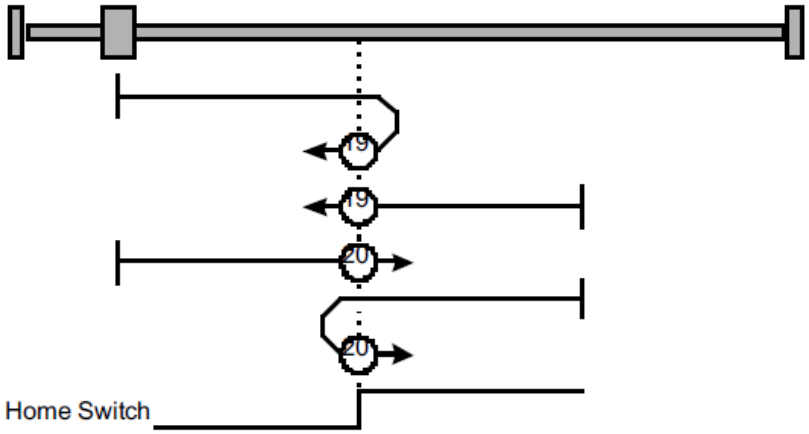
（2） Use negative limit switch

If this method 11~14 is almost same as method 7~10,the drive first moves relatively quick into the negative direction.



Method 17~20, 23~30: Homing operation to the negative limit switch

If this method is used the drive first moves relatively quick into the negative direction,until it reaches the negative limit switch. This is displayed in the diagram by the rising edge. Afterwards the drive slowly returns and searches for the exact position of the limit switch. The zero position refers to the descending edge from the negative limit switch.



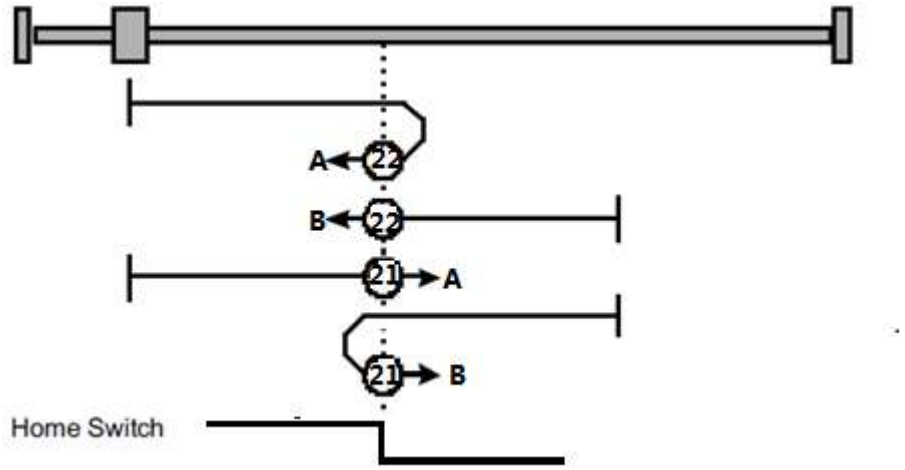
Methods 21,22 Using reference switch

Methods 21 A: When homing mode is enabled, If reference switch H-S =1, the drive first moves slowly into the positive direction until reaches the falling edge of H-S.

Methods 21 B: When homing mode is enabled, If reference switch H-S=0, the drive first moves relatively quick into the negative direction until it reaches the reference switch This is displayed in the diagram by the rising edge. Then the drive returns slowly to the positive derection, stops until reaches the falling edge of the H-S.

Methods 22 A: When homing mode is enabled, If reference switch H-S=1, the drive first moves relatively quick into the positive direction until it reaches the reference switch. This is displayed in the diagram by the falling edge. Afterwards the drive slowly returns, and stops until reaches the rising edge of the H-S.

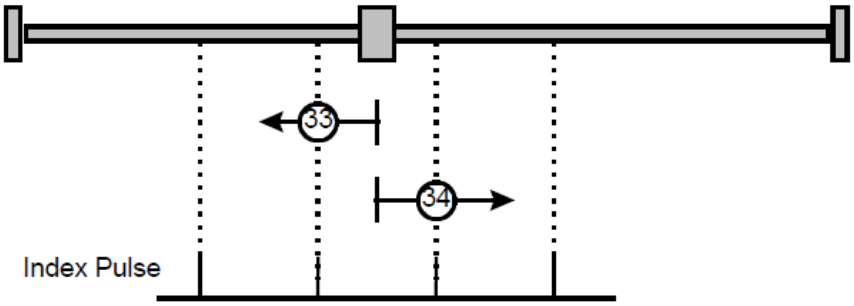
Methods 22 B: When homing mode is enabled, If reference switch H-S =0, the drive first moves slowly into the negative direction until reaches the rising edge of the H-S.



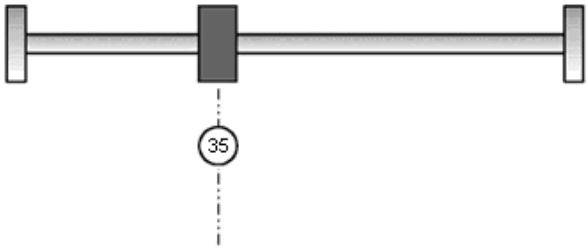
Methods 33,34 Using zero impulse evaluation

Methods 33: The drive moves slowly into the negative direction, stops until reaches the zero impulse evaluation.

Methods 34: The drive moves slowly into the positive direction, stops until reaches the zero impulse evaluation.

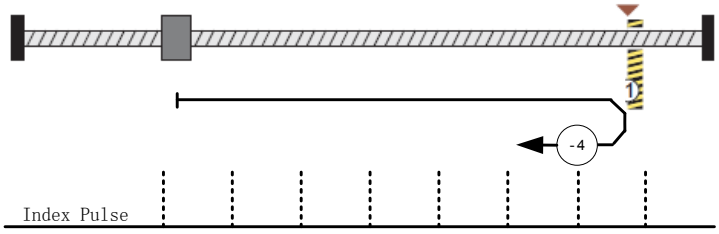


Method 35: Set current position as the homing point



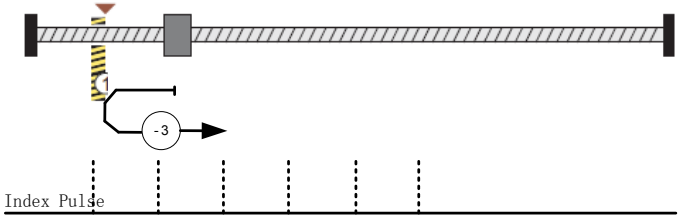
Method -4 Movement in positive direction, hitting an end and reversing to travel，the target homing position is the first C pulse

In this method, the motor moves in positive direction. When it hits an end so that the torque set in Pn207 is reached for the blocking time set in Pn208,movement in the opposite direction, and the target homing position is the first C pulse.



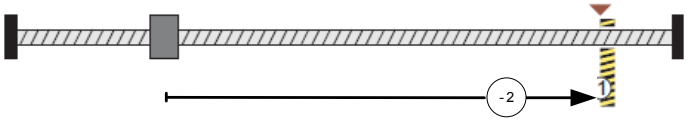
Method -3 Movement in negative direction, hitting an end and reversing to travel，the target homing position is the first C pulse

In this method, the motor moves in negative direction. When it hits an end so that the torque set in Pn207 is reached for the blocking time set in Pn208,movement in the opposite direction, and the target homing position is the first C pulse.



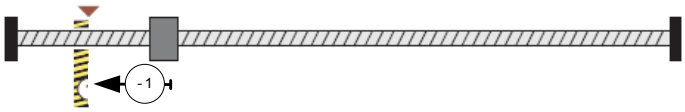
Method -2:Movement in positive direction, hitting an end, makes the current position for the origin.

In this method, the motor moves in positive direction. When the drive hits an end so that the torque set in Pn207 is reached for the blocking time set in Pn208,and makes the current position for the origin.



Method -1:Movement in negative direction, hitting an end, makes the current position for the origin.

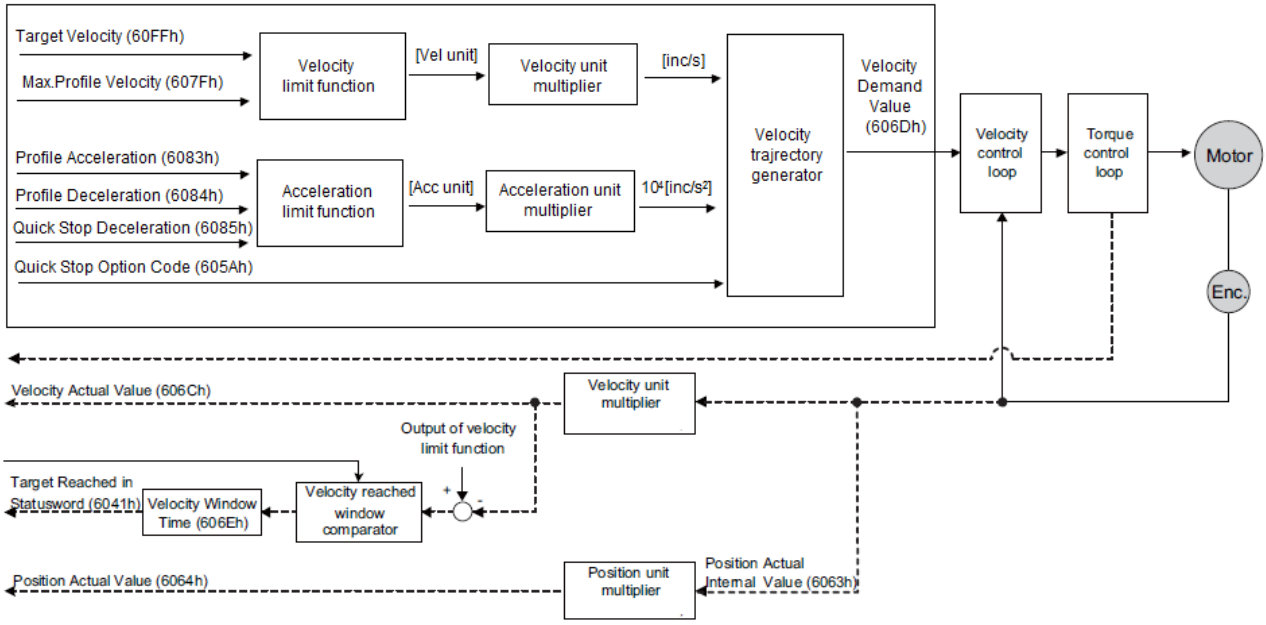
In this method, the motor moves in negative direction. When the drive hits an end so that the torque set in Pn207 is reached for the blocking time set in Pn208,and makes the current position for the origin.



Notes: When starting homing on homing methed about input signal, the rotation direction of servo motor is associated with the initial status of the input signal.Changing the inicial status by inverse input on set Pn516/Pn517 if it is necessary.When using reference switch homing,I/O should be set as C:HmRef by Pn509/Pn510.

7.3 Profile velocity mode

7.3.1 Flow diagram of profile velocity mode



7.3.2 Control word of profile velocity mode

15 ~ 9	8	7 ~ 4	3 ~ 0
*	Halt	*	*

*: referred to previous chapters

Name	Value	Description
Halt	0	Execute the motion
	1	Stop axle

7.3.3 Status word of velocity mode

15 ~ 14	13	12	11	10	9 ~ 0
*	MaxSlippageError	Speed	*	Target reached	*

*: Referred to previous chapters

Name	Value	Description
Target reached	0	Halt = 0: *Target velocity* not (yet) reached Halt = 1: Axle decelerates
	1	Halt = 0: *Target velocity* reached Halt = 1: Axle has velocity 0
Speed	0	Speed is not equal 0
	1	Speed is equal 0
Max slippage error	0	Maximum slippage not reached
	1	Maximum slippage reached

7.3.4 Relevant parameters of profile velocity mode

Index	Object	Name	Type	Attr.
6069 h	VAR	velocity_sensor_actual_value	INT32	RO
606B h	VAR	velocity_demand_value	INT32	RO
606C h	VAR	velocity_actual_value	INT32	RO
606D h	VAR	velocity_window	UINT16	RW
606E h	VAR	velocity_window_time	UINT16	RW
606F h	VAR	velocity_threshold	UINT16	RW
6070 h	VAR	velocity_threshold_time	UINT16	RW
607F h	VAR	Max profile velocity	UINT32	RW
60FF h	VAR	target_velocity	INT32	RW

velocity_sensor_actual_value

The speed encoder is read via the object **velocity_sensor_actual_value**. The value is normalised in internal units. The velocity demand value can be read via this object.

Index	6069 h
Name	velocity_sensor_actual_value
Object Code	VAR
Data Type	INT32
Access	RW
PDO Mapping	YES
Units	0.1rmps (1R/10min)
Value Range	--
Default Value	--

velocity_demand_value

The velocity demand value can be read via this object. The unit of this object is the unit of user's speed unit. The velocity demand value can be read via this object.

Index	606B h
Name	velocity_demand_value
Object Code	VAR
Data Type	INT32
Access	RO
PDO Mapping	YES
Units	speed units
Value Range	--
Default Value	--

velocity_actual_value

The actual velocity value can be read via the object **velocity_actual_value**。The velocity demand value can be read via this object.

Index	606C h
Name	velocity_actual_value
Object Code	VAR
Data Type	INT32
Access	RO
PDO Mapping	YES
Units	speed units
Value Range	--
Default Value	--

velocity_window

With the object **velocity_window** a tolerance window for the velocity actual value will be defined for comparing the **velocity_actual_value**（606C h） with the target velocity (**target_velocity** object **60FFh**). If the difference is smaller than the velocity window （606D h） for a longer time than specified by the object **velocity_window_time** （606E h） bit 10 (**target_reached)** will be set in the object **statusword**.

Index	606D h
Name	velocity_window
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	--
Default Value	20 R/10min

velocity_window_time

The object **velocity_window_time** serves besides the object **606Dh**: **velocity_window** to adjust the window comparator.

Index	606E $_{h}$
Name	velocity_window_time
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	YES
Units	ms
Value Range	--
Default Value	0

velocity_threshold

The object **velocity_threshold** determines the velocity underneath the axis is regarded as stationary. As soon as the **velocity_actual_value** exceeds the **velocity_threshold** longer than the **velocity_threshold_time** bit 12 is cleared in the **statusword**.

Index	606F $_{h}$
Name	velocity_threshold
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	--
Default Value	10 R/10min

velocity_threshold_time

The object **velocity_threshold** determines the velocity below the axis is regarded as stationary. Its unit is ms.As soon as the **velocity_actual_value** exceeds the **velocity_threshold** longer than the **velocity_threshold_time** bit 12 is cleared in the **statusword**.

Index	6070 $_{h}$
Name	velocity_threshold_time
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	YES
Units	ms
Value Range	--
Default Value	0

Max profile velocity

The object max profile velocity is the speed that the motor can not exceed. Its unit is the unit of customer's speed.

Index	607F $_{h}$
Name	Max profile velocity
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	--
Default Value	0

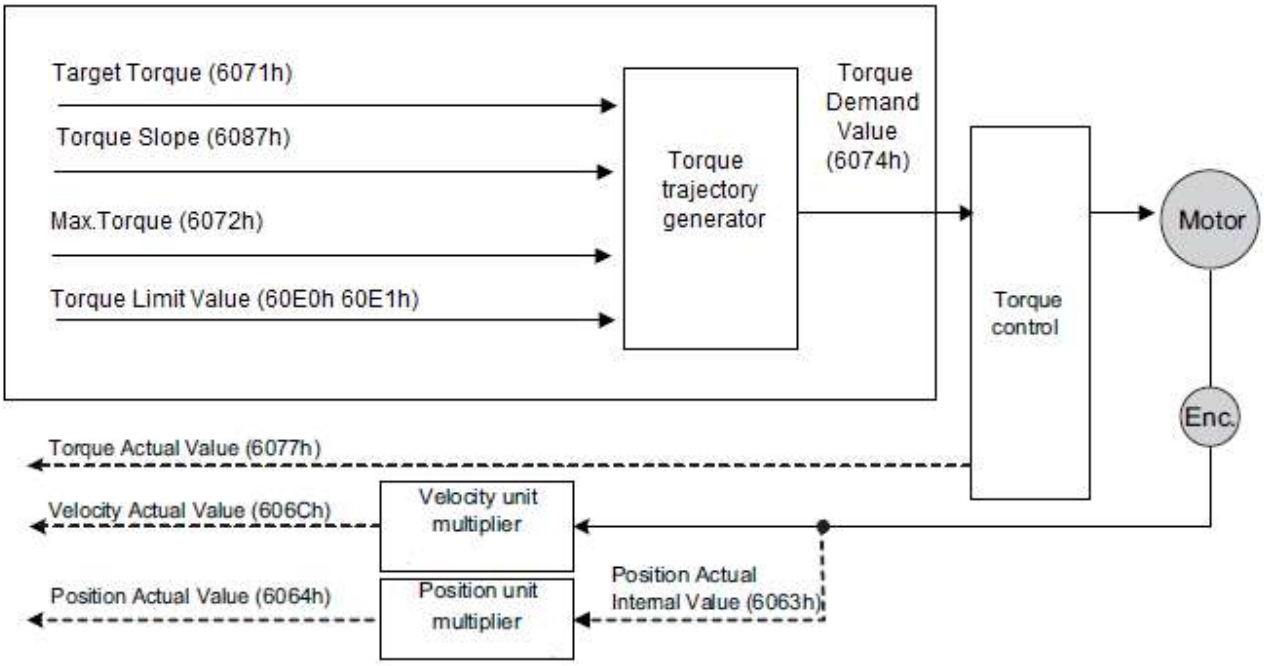
target_velocity

The object **target_velocity** is the setpoint for the ramp generator.

Index	60FF $_{h}$
Name	target_velocity
Object Code	VAR
Data Type	INT32
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	--
Default Value	0

7.4 Profile torque mode

7.4.1 Flow diagram of profile torque mode



7.4.2 Control word of profile torque mode

15 ~ 9	8	7 ~ 4	3 ~ 0
*	Halt	*	*

*: referred to previous chapters

bit	Value	Definition
8	0	The motion shall be executed or continued
	1	Axis shall be stopped according to the halt option code (605Dh)

7.4.3 Status word of profile torque mode

15 ~ 14	13	12	11	10	9 ~ 0
*	*	*	*	Target reached	*

*: referred to previous chapters

bit	Value	Definition

10	0	Target torque not reached
	1	Target torque reached

7.4.4 Relevant parameters of profile torque mode

Index	Object	Name	Type	Attr.
6071 h	VAR	target_torque	INT16	RW
6072 h	VAR	Max torque	UINT16	RW
6074 h	VAR	torque_demand	INT16	RO
6077 h	VAR	torque_actual_value	INT16	RO
6087 h	VAR	torque_slope	UINT32	RW

target_torque

The torque command can be sent via target_torque. Unit：0.1% rated torque

Index	6071 h
Name	target_torque
Object Code	VAR
Data Type	INT16
Access	RW
PDO Mapping	YES
Units	0.1% rated torque
Value Range	--
Default Value	--

Max torque

The object max torque is the torque that the motor can not exceed. Unit：0.1% rated torque

Index	6072 h
Name	Max torque
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	YES
Units	0.1% rated torque
Value Range	--
Default Value	0

torque_demand

The output of the torque command generator.The driver generates the command according to the value of Target_Torque and Torque_Slope.

Index	6074 h
Name	torque_demand

Object Code	VAR
Data Type	INT16
Access	RO
PDO Mapping	YES
Units	0.1% rated torque
Value Range	--
Default Value	--

torque_actual_value

The torque output can be read via torque_actual_value. Unit：0.1% rated torque

Index	6077 h
Name	torque_actual_value
Object Code	VAR
Data Type	INT16
Access	RO
PDO Mapping	YES
Units	0.1% rated torque
Value Range	--
Default Value	--

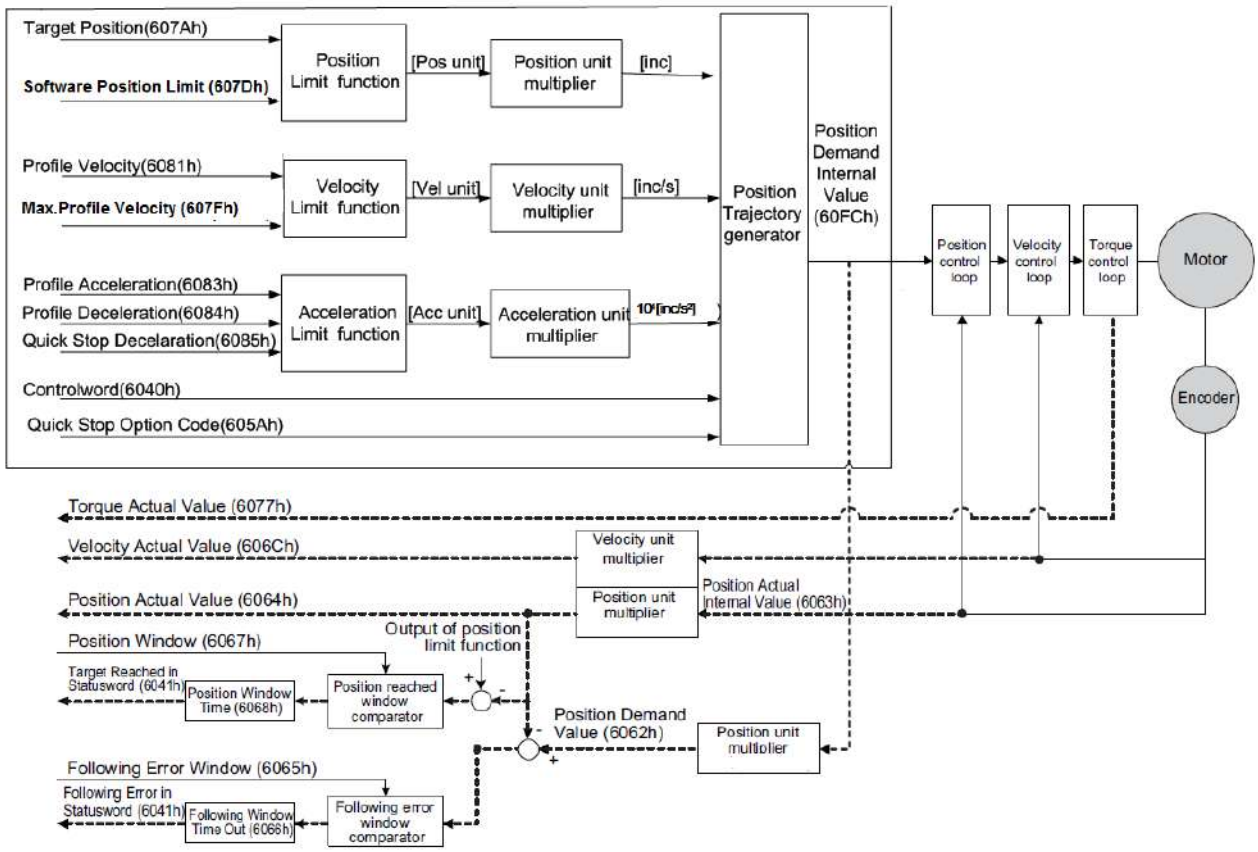
torque_slope

The speed of variational torque command can be set via torque_slope. Unit：0.1% rated torque /S

Index	6087 h
Name	torque_slope
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	0.1% rated torque /S
Value Range	--
Default Value	--

7.5 Profile position mode

7.5.1 Flow diagram of profile position mode



7.5.2 Control word of profile position mode

15 ~ 9	8	7	6	5	4	3 ~ 0
*	Halt	*	abs / rel	Change set immediately	New set-point	*

*: referred to previous chapter

Name	Value	Description
New set-point	0	Does not assume *target position*
	1	Assume *target position*
Change set immediately	0	Finish the actual positioning and then start the next positioning
	1	Interrupt the actual positioning and start the next positioning
abs / rel	0	*Target position* is an absolute value
	1	*Target position* is a relative value
Halt	0	Execute positioning
	1	Stop axle with *profile deceleration* (if not supported with *profile acceleration*)

7.5.3 Status word of profile position mode

15 ~ 14	13	12	11	10	9 ~ 0
*	Following error	Set_point acknowledge	*	Target reached	*

*: referred to previous chapter

Name	Value	Description
Target reached	0	Halt = 0: *Target position* not reached Halt = 1: Axle decelerates
	1	Halt = 0: *Target position* reached Halt = 1: Velocity of axle is 0
Set-point acknowledge	0	Trajectory generator has not assumed the positioning values (yet)
	1	Trajectory generator has assumed the positioning values
Following error	0	No following error
	1	Following error

7.5.4 Revelant parameters of profile position mode

Index	Object	Name	Type	Attr.
607A $_h$	VAR	target_position	INT32	RW
6081 $_h$	VAR	profile_velocity	UINT32	RW
6082 $_h$	VAR	end_velocity	UINT32	RW
6083 $_h$	VAR	profile_acceleration	UINT32	RW
6084 $_h$	VAR	profile_deceleration	UINT32	RW
6085 $_h$	VAR	quick_stop_deceleration	UINT32	RW
6086 $_h$	VAR	motion_profile_type	INT16	RW
60A4-01 h	VAR	Profile_jerk1	UINT32	RW

target_position

The object **target_position** determines the destination the servo controller moves to. The target position (**target_position**) is interpreted either as an absolute or relative position. This depends on bit 6 (**relative**) of the object **control word**.

Index	607A $_h$
Name	target_ position
Object Code	VAR
Data Type	INT32
Access	RW
PDO Mapping	YES
Units	position units
Value Range	--
Default Value	0

profile_velocity

The object **profile_velocity** specifies the speed that usually is reached during a positioning motion at the end of the acceleration ramp. The object **profile_velocity** is specified in **speed_units**.

Index	6081 $_h$
Name	profile_velocity
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	--
Default Value	0

end_velocity

The object **end_velocity** defines the speed at the target position (**target_position**). Usually this object has to be set to zero so that the controller stops when it reaches the target position. For gapless sequences of positionings a value unequal zero can be set.

Index	6082 h
Name	end_velocity
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	speed units
Value Range	--
Default Value	0

profile_acceleration

The object **profile_acceleration** determines the maximum acceleration used during a positioning motion. It is specified in user specific acceleration units (**acceleration_units)**.

Index	6083 h
Name	profile_acceleration
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	acceleration units
Value Range	--
Default Value	100000 R/10min/s

profile_deceleration

The object **profile_deceleration** specifies the maximum deceleration used during a positioning motion. This object is specified in the same units as the object **profile_acceleration**

Index	6084 h
Name	profile_deceleration
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	acceleration units
Value Range	--
Default Value	100000 R/10min/s

quick_stop_deceleration

The object **quick_stop_deceleration** determines the deceleration if a Quick Stop will be executed.

Index	6085 h
Name	quick_stop_deceleration
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	acceleration units
Value Range	--
Default Value	200000 R/10min/s

motion_profile_type

The object **motion_profile_type** is used to select the kind of speed profile. At present only a linear trpezia profile(set as 0) and a stable S linear jerk profile are available(set as 2).

Index	6086 h
Name	motion_profile_type
Object Code	VAR
Data Type	INT16
Access	RW
PDO Mapping	YES
Units	--
Value Range	0
Default Value	0

profile_jerk1

profile_jerk1 is used to set the jerk of speed profile. The value is more smaller,the speed changing is more smooth.

Index	60A4 -01h
Name	profile_jerk1
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Units	jerk units
Value Range	1-20
Default Value	5pulse/(s*100μs*100μs)

7.5.5 Function description

When the speed profile is trapezia(motion_profile_type=0),two different ways to apply target positions to the servo controller are supported.

Single setpoints

After reaching the **target_position** the servo controller signals this status to the host by the bit **target_reached** (Bit 10 of **controlword**) and then receives a new setpoint. The servo controller stops at the **target_position** before starting a move to the next setpoint.

When moving to a setpoint, the servo controller signals a new setpoint by the bit **target_reached** (Bit 4 of **controlword**).Than the servo drive will move to the new setpoint.

Set of setpoints

After reaching the **target_position** the servo controller signals this status to the host by the bit **target_reached** (Bit 10 of **controlword**) and then receives a new setpoint. The servo controller stops at the **target_position** before starting a move to the next setpoint.

These Two methods are controlled by the bit4 and **bit5** in the object **controlword** and **set_point_acknowledge** in the object **statusword**. These bits are in a request-response relationship. So it is possible to prepare one positioning job while another job is still running.

Simple job positioning:

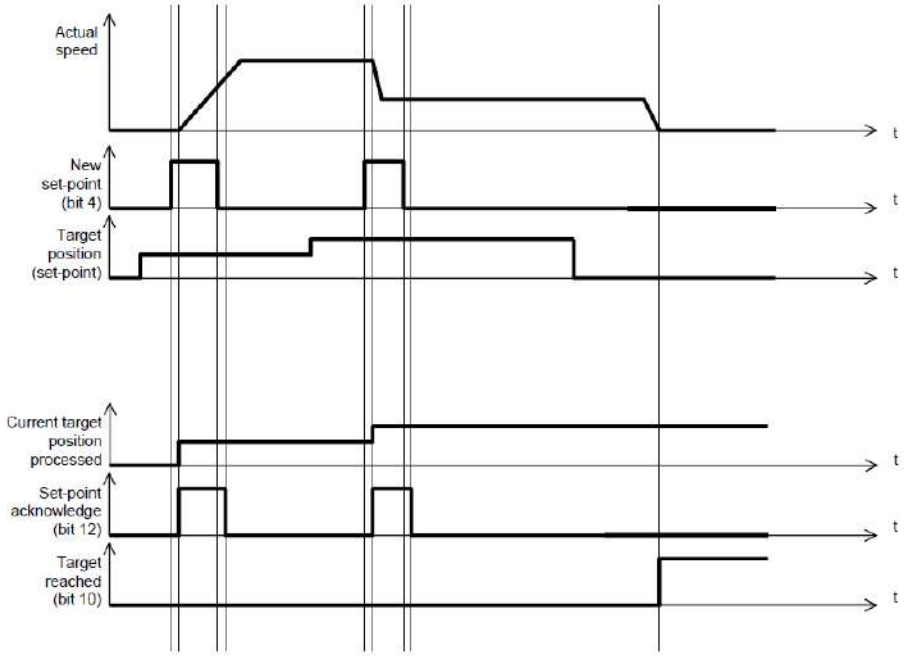
1.At first set NMT as Operational and control mode parameter (6061h) as 1.

2. At first the positioning data (**target_position:** 607A$_h$ **, profile_velocity, end_velocity** and **profile_acceleration**) are transferred to the servo controller.

3. The host can start the positioning motion by setting the bit4 (**new_set_point)** in the **controlword** as 1, bit5 (change_set_immediately) as 1 and bit6 as absolute or referential type according to target position type (absolute or referential).

4. This will be acknowledged by the servo controller by setting the bit **set_point_acknowledge** in the **statusword** when the positioning data has been copied into the internal buffer. Motion could be started now.

5. When the target is reached, drive will be acknowledged by bit 10 (target_reached) in status word. And then it will run gapless according to program or accept a new target position.



Gapless sequence of positioning job:

1.At first set NMT as Operational and control mode parameter (6061h) as 1.

2. At first the positioning data (**target_position:** 607A$_h$**, profile_velocity, end_velocity** and **profile_acceleration**) are transferred to the servo controller.

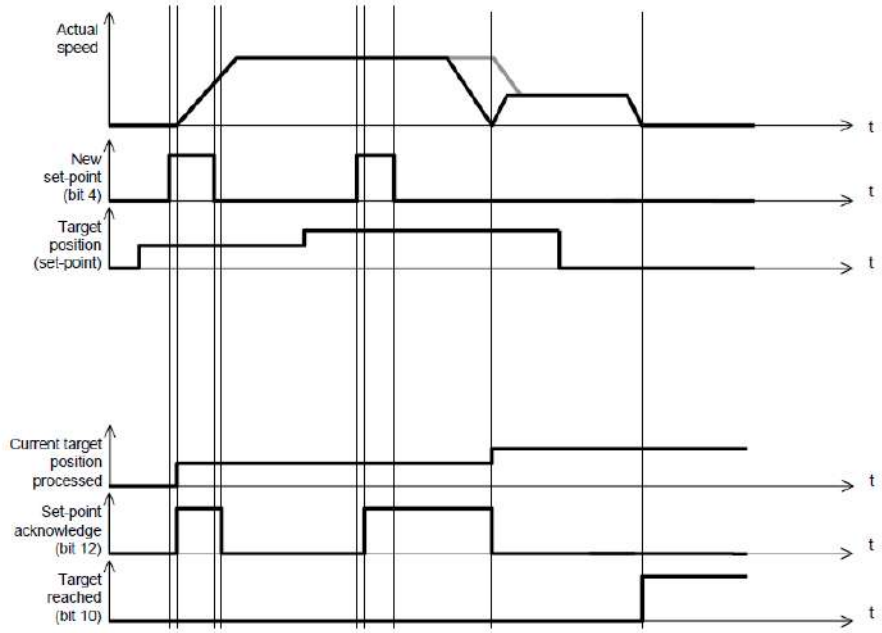
3. The host can start the positioning motion by setting the bit4 (**new_set_point)** in the **controlword** as 1, bit5 (change_set_immediately) as 0 and bit6 as absolute or referential type according to target position type (absolute or referential).

4. This will be acknowledged by the servo controller by setting the bit **set_point_acknowledge** in the **statusword** when the positioning data has been copied into the internal buffer. Motion could be started now.

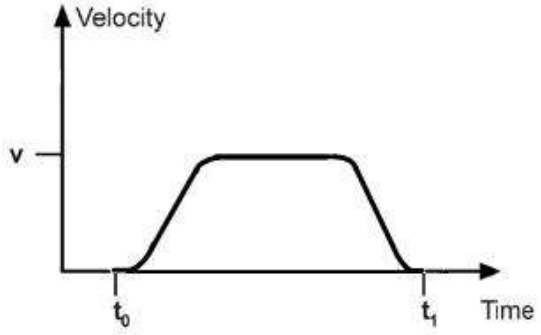
5.Second positioning data (**target_position:** 607A$_h$**, profile_velocity, end_velocity** and **profile_acceleration**) are transferred to the servo controller.

6. The host can start the positioning motion by setting the bit4 (**new_set_point)** in the **controlword** as 1, bit5 (change_set_immediately) as 0 and bit6 as absolute or referential type according to target position type (absolute or referential).

7. When the 1 target is reached driver will move forward to second target position. When the second target position is reached drive will be acknowledged by bit10 (target reached) in status word. And then it will be executed by program or accept another new target position.

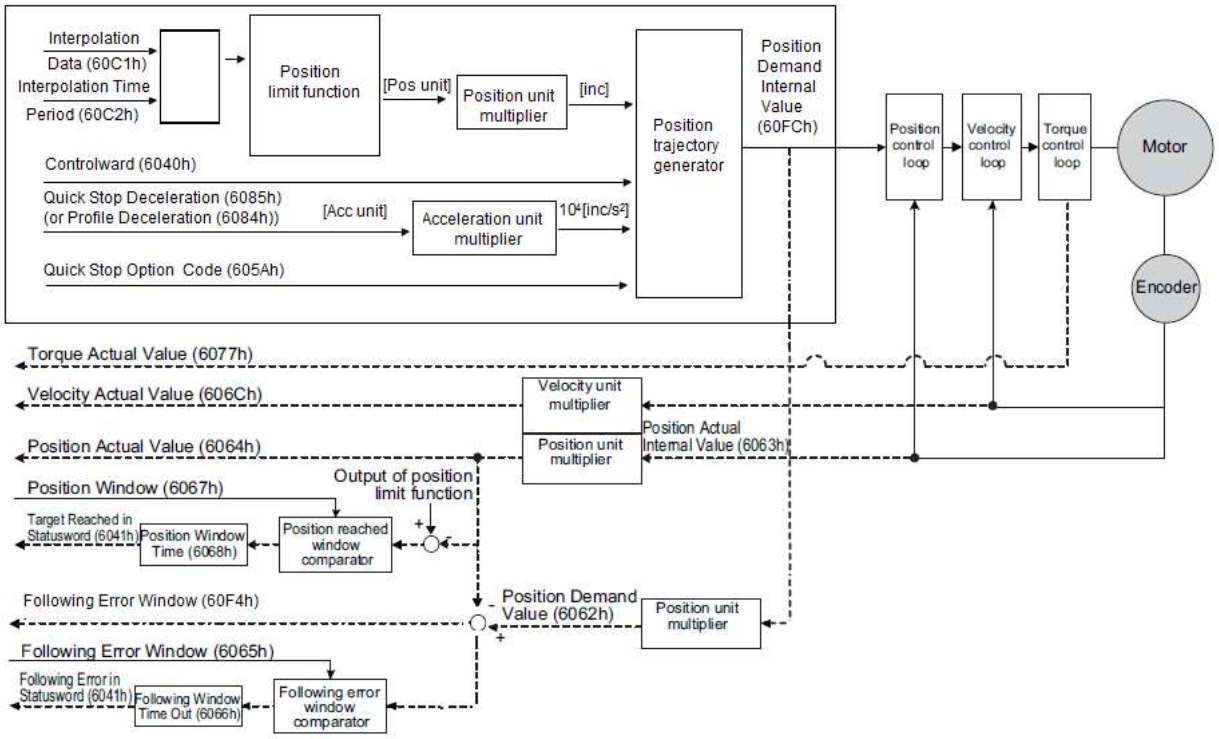


When the speed profile is S (motion_profile_type=2), only set of setpoints is available. 6083 h (profile_acceleration) limits max. acceleration. 6081h(profile_velocity) limits max.speed. 60A4-01 h (VAR Profile_jerk1) limits the jerk.now only symmetrical S linear is available.



7.6 Interpolation position mode

7.6.1 Flow diagram of interpolation position mode



7.6.2 Control word of interpolation position mode

15 ~ 9	8	7	6	5	4	3 ~ 0
*	Halt	*	*	*	Enable ip mode	*

*: Please refer to the chapters ahead

Name	Value	Description
Enable ip mode	0	Interpolated position mode inactive
	1	Interpolated position mode active
Halt	0	Execute the instruction of bit 4
	1	Stop axle

7.6.3 Status word of interpolation position mode

15 ~ 14	13	12	11	10	9 ~ 0
*	*	ip mode active	*	Target reached	*

*: Please refer to the chapters ahead

Name	Value	Description
Target reached	0	Halt = 0: Position not (yet) reached Halt = 1: Axle decelerates
	1	Halt = 0: Position reached Halt = 1: Axle has velocity 0
ip mode active	0	Interpolated position mode inactive
	1	Interpolated position mode active

7.6.4 Parameters of position interpolation control

Index	Object	Name	Type	Attr.
60C0 $_h$	VAR	Interpolation sub mode select	INT16	RW
60C1 $_h$	ARRAY	Interpolation data record	INT32	RW
60C2 $_h$	RECORD	Interpolation time period		RW

Interpolation sub mode select

Interpolation sub mode select is used to select the method of interpolation under IP control. Pronet servo drive only offers linear interpolation.

Index	60C0h
Name	Interpolation sub mode select
Object Code	VAR
Data Type	INT16
Access	RW
PDO Mapping	NO
Value Range	0
Default Value	0
Comment	0: Linear interpolation

Interpolation data record

Interpolation data record is used to reserve interpolation potion data. Our servo drive's interpolation command only uses the first data whose subindex is 1.

Index	60C1h
Subindex	0

Object Code	ARRAY
Data Type	INT32
Access	RO
PDO Mapping	YES
Value Range	INT8
Default Value	2
Comment	number of entries

Index	60C1h
Subindex	1
Object Code	ARRAY
Data Type	INT32
Access	RW
PDO Mapping	YES
Value Range	INT32
Default Value	0
Comment	the first parameter of ip function

Index	60C1h
Subindex	2
Object Code	ARRAY
Data Type	INT32
Access	RW
PDO Mapping	YES
Value Range	INT32
Default Value	0
Comment	The second parameter of ip function

Interpolation time period

Interpolation time period is used to reserve the time data of interpolation position.

Index	60C2h
Object Code	RECORD
Data Type	Interpolation time period record (0080h)
Category	Conditional: mandatory if ip, csp, csv or cst mode is supported

Index	60C2h
Subindex	0
Object Code	RECORD
Data Type	UINT8
Access	C

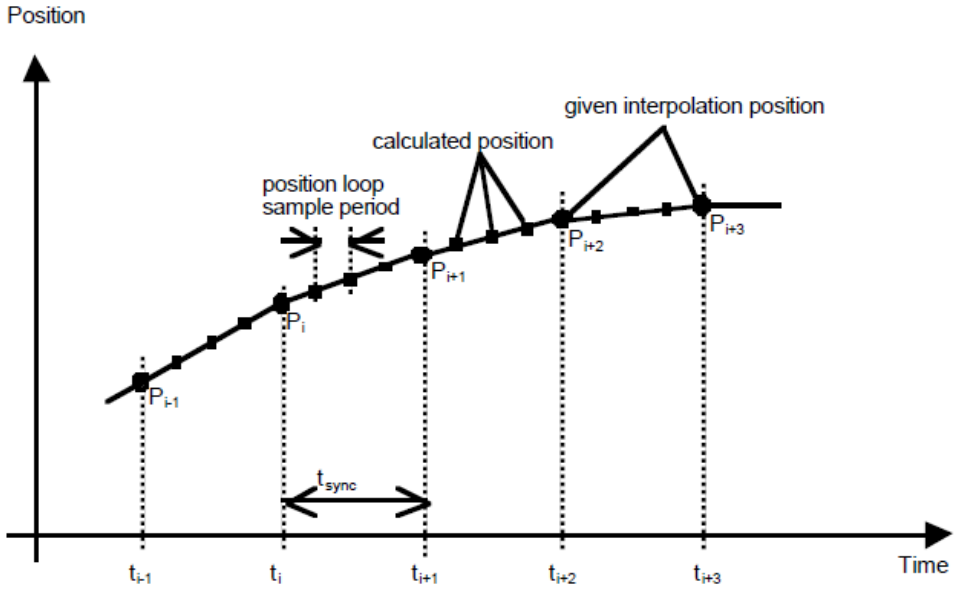
PDO Mapping	NO
Value Range	02
Default Value	02
Comment	Highest sub-index supported

Index	60C2h
Subindex	01
Object Code	RECORD
Data Type	UINT8
Access	RW
PDO Mapping	YES
Value Range	UINT8
Default Value	01
Comment	Interpolation time period value

Index	60C2h
Subindex	02
Object Code	RECORD
Data Type	INT8
Access	RW
PDO Mapping	YES
Value Range	-128 to +63
Default Value	-3
Comment	Interpolation time index

7.6.5 Function description

Interpolation principle in IP mode:



Pi: interpolation position set by the host

t_{sync}: sync period

Some hints:

1. In our servo drive, there is no buffer for position data so in IP control, all the position data needs to be updated by the controller. To achieve synchronization, controllers need to send the updated position at first and then use SYNC signal to make all the servo drive receive the synchronization information. After receiving the synchronization information, servo drive will synchronize its internal clock. Please notice that the sync period should be not bigger than interpolation cycle period in order to keep the updating of interpolation data.
2. In IP mode, the host should at first set the servo's PDO receiving method into sync mode (Use SYNC frame to receive and send synchronization information). Because SYNC is broad casted, every servo drive will only update PDO data after receiving this signal.
3. Before SYNC is sent, we need host to send position data Xi and control word to the servo drive.
4. When there is data delay, servo drive will use the last sync date to do interpolation.
5. After one sync period, if there is no further data updating, interpolation cycle overtime alarm （A 69）will happen. And then servo drive will stop.

Recommended RPDO configuration:

When you use only one RPDO,

Control word(index:6040h,subindex:0h)	32bit position reference (index:60C1h,subindex:01h)

When you use two RPDO,

Control word(index:6040h,subindex:0h)

32bit position reference (index:60C1h,subindex:01h)

Configuration process:

1. Configure PDO. (RPDO1 is configured as index: 6040h, subindex: 0h, RPDO2 is configured as index 60c1h, subindex: 1h)

2. Set interpolation cycle time 2105h and 60C2, the unit is micro send (us). Please notice that both values need to be configured. For example, if the cycle time is 2ms, you need to set 2105h as 2000 and 60c2:01 as 2, 60c2:02 as -3.

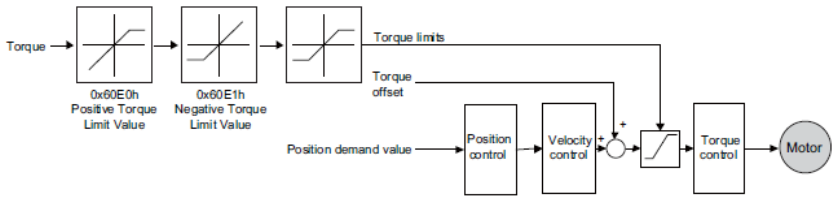
3. Set sync cycle time (1006h), the unit is micro send (us)

4. Set PDO as Sync mode (Set the object dictionary (index: 1400h, subindex: 02h) as 1. Set object dictionary (index: 1401h, subindex: 02h) as 1). If sending PDO needs to be in sync mode as well, we need to set object dictionary (index: 1800h, subindex: 02h) as 1 and (index: 6060h, subindex: 0h) as 1 as well.

5. NMT starts node.

7.7 Torque limit function

In CANopen bus mode,torque limit function is realized by 0x60E0 snd 0x60E1 as below.



PosTorLimit(0x60E0):positive torque limit,unit: 0.1% rated torque

Index	60E0h
Name	PosTorLimit
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	YES
Value Range	0-3000
Default Value	3000

NegTorLimit(0x60E1): negative torque limit,unit: 0.1% rated torque

Index	60E1h
Name	NegTorLimit
Object Code	VAR
Data Type	UINT16
Access	RW
PDO Mapping	YES
Value Range	0-3000
Default Value	3000

Chapter 8 Parameters of the CAN interface

Parameter	Name and discription	Reboot required	Available for which control method	Functions and content
Pn006	**Hexadecimal**	required	ALL	**Pn006.0 Bus type selection** [0]No bus [1]PROFIBUS-DP V0/V1 [2]PROFIBUS-DP V2 [3] CANopen **Pn006.1 Reserved** **Pn006.2 Low-frequency vibration suppression switch** [0]Low-frequency vibration suppression function disabled [1]Low-frequency vibration suppression function enabled **Pn006.3 Reference input filter for open collector signal** [0] when pulse is difference input, servo receiving pulse frequency ≤4M [1] when pulse is difference input, servo receiving pulse frequency ≤650K [2] when pulse is difference input, servo receiving pulse frequency ≤150K
Pn703	**Hexadecimal**	required	ALL	**Pn703.0** CANopen **baud rate** [0] 50Kbps [1] 100Kbps [2] 125Kbps [3] 250Kbps [4] 500Kbps [5] 1Mbps **Pn703.1 Reserved** **Pn703.2 Reserved** **Pn703.3 Reserved**
Pn704	**Axis address**	required	ALL	CANopen axis address

Notes:EDS,ETS and ProNet use the same node.These parameters is appropriate for each axis of EDS and ETS.

Chapter 9 CAN communication example

The entire test below is based on three conditions:

1. Communication has been established correctly.
2. The address of the servo drive is 1.
3. All the message data is hexadecimal.

9.1 SDO configuration

SDO operation is to read and write parameters (0601h → host sends 0581h →slave sends)

Address: 3022h（Pn118）. Write 1000. And then read this parameter.

Activate the downloading process: 2B，3022，00，03E8

That is …

601(ID) 2B 22 30 00 E8 03 00 00

The servo drive should respond 60，3022，00，00，00，00，00

That is 581(ID) 60 22 30 00 00 00 00 00

Activate the uploading: 40，3022，00，0000

That is 601(ID)40 22 30 00 00 00 00 00

The servo drive needs to respond: 43，3022，00，03E8

That is: 581 (ID) 43 22 30 00 E8 03 00 00

9.2 PDO configuration

RPDO mapping example: To configure two RPDO, one of which is 6040h and the other are 607A and 6081h) . The slaves respond of message 581h(ID) is omitted.

RPDO MAPPing

601 (ID)2F 00 16 00 00 00 00 00 //RPDO1 stop

First RPDO 201

601 (ID)23 00 16 01 10 00 40 60 //6040h

601(ID) 2F 00 16 00 01 00 00 00 // RPDO1 enable

601(ID) 2F 01 16 00 00 00 00 00 //RPDO2 stop

Second RPDO 301

601(ID) 23 01 16 01 20 00 7A 60 //607Ah and 6081h

601 (ID)23 01 16 02 20 00 81 60

601(ID) 2F 01 16 00 02 00 00 00// RPDO2 enable

And then set the transmit PDO as SYNC or Timing method. The default setting is Time method.

After configuring the PDO, if you need to activate the configuration, you need to reset the communication.
NMT is OPERATIONAL: 00 01 01//（the first "01" is the start node instruction,the second "01" is the number of the node）

Attention：
1.Before configuration, please stop PDO. For example, Cleaning the value with index 1600h and sub-index 00, cleaning the value to 0 is necessary). After configuration, please set a correct number of PDO(For example, set the value with index 1600h and sub-index 00 as 1) to activate the PDO.
2. Please pay attention to the data length and number. Wrong setting will lead to wrong configuration.
3. Please mapping time key object to PDO according to the actual needs to decrease bus load.

9.3 Profile position mode

At first, please mapping and configure PDO according to the example above and activate the communication.
And then, please set the control mode.
message：601(ID) 2F 60 60 00 01 00 00 00//set 6060h as 1（position contrl is PP）
And then, set status machine as Operation Enable
message：601(ID) 2B 40 60 00 06 00 00 00//set 6040h as 6,switch to "ready to switch on"
message：601(ID) 2B 40 60 00 07 00 00 00 //set 6040h as 7, switch to "switch on"
message：601(ID) 2B 40 60 00 0F 00 00 00 //set 6040h as F, switch to "operation enable" and servo-on；

Note: this switching flow is based on successful switching received command and original state is "switch on disabled". "status word" should be read in the practice.please refer to **6.1 State diagram (State machine), 6.2.1Controlword** and **6.2.2 Statusword.**

And then, send data by PDO
Let servo motor rotate for 5 revolutions (Set PDO1 as 6040(status word), PDO2 as 607A(position pulse number) and 6081(velocity, unit as much as 0.1rpm)

Send RPDO2 The data is as below …
message：301 (ID) 50 C3 00 00 2C 01 00 00(50000,300)// 50 C3 00 00 is position data，that is, 50000 pulses；2C 01 00 00 is speed，that is, 30rpm；

Send RPDO1 as below
1、message：201(ID) 0F 00 //；Clear the bit4 of 6040 as 0.
2、message：201(ID) 1F 00 // Clear the bit4 of 6040 as 1 and servo motor is operating under absolute position; Motor runs.
3、message：201(ID) 0F 00 //Clear the bit4 of 6040.
4、message：201(ID) 5F 00 // Clear the bit4 of 6040 as 1. The servo motor runs under incremental position.
5、message：201(ID) 0F 00 //Clear bit4 of 6040 as 0.

Attention:
1) The servo drive is using ↑of 6040's bit 4 to accept new position order. So after every single operation, the bit needs to be cleared. Host needs to check bit12 of status word 6040 in the servo drive to decide whether or not to give new data to servo systems. When status word 6041 in the servo drives 0, it means the servo drive is ready for new data and order. If the value is 1, the order won't be executed even if there is data for the servo drive to receive.

2) In absolute approach, continuous position updating is required.

If you want to change the operating distance, you need to send RPDO2 again.
RPDO2:
301 B0 3C FF FF 2C 01 00 00 (-50000,-300)//That is, -50000 pulses; 30rpm.

9.4 Two-axis interplate position mode

At first, mapping and configure PDO
/receive 2 PDO by default：RPDO1：60C1h--01h
// Send two PDO by default：TPDO1：6041h TPDO2：6064h/606Ch
// pulse，Velocity 0.1rpm

Configure 1 RPDO
RPDO MAPPing
message：601(ID) 2F 00 16 00 00 00 00 00 //RPDO1 stop
message：601(ID) 23 00 16 01 20 01 C1 60 //60C1h,sub01
message：601(ID) 2F 00 16 00 01 00 00 00// RPDO1 enable

Configure 2 TPDO，TPDO1：6041h TPDO2：6064h/606Ch
TPDO MAPPing
message：601(ID) 2F 00 1A 00 00 00 00 00 //TPDO1 stop
message：601(ID) 23 00 1A 01 10 00 41 60 //6041h
message：601(ID) 2F 00 1A 00 01 00 00 00 // TPDO1 enable

message：601(ID) 2F 01 1A 00 00 00 00 00 //RPDO2 stop
message：601(ID) 23 01 1A 01 20 00 64 60 //6064h and 606Ch
message：601(ID) 23 01 1A 02 20 00 6C 60 //
message：601(ID) 2F 01 1A 00 02 00 00 00// TPDO2 enable

Set Sync time.
message：601(ID) 2F C2 60 01 10 00 00 00 //60C2h-01----->1ms，set according to the actual needs
message：601(ID) 2F C2 60 02 FD 00 00 00 //

Configure the PDO receiving and sending are both activated by one Sync frame.
Set 1400h
message：601(ID) 2F 00 14 02 01 00 00 00 //1400-02---->SYNC
Set 1800h
message：601(ID) 2F 00 18 02 01 00 00 00 //1800-02---->SYNC
Set 1801h
message：601(ID) 2F 01 18 02 01 00 00 00 //1801-02---->SYNC

Set control mode
message：601(ID) 2F 60 60 00 07 00 00 00//Set 6060h as7 (IP position control)
And then, set the status machine

message: 601(ID) 2B 40 60 00 06 00 00 00// Set 6040h as 6
message: 601(ID) 2B 40 60 00 07 00 00 00 // Set 6040h as 7
message: 601(ID) 2B 40 60 00 0F 00 00 00 // Set 6040h as F to servo on

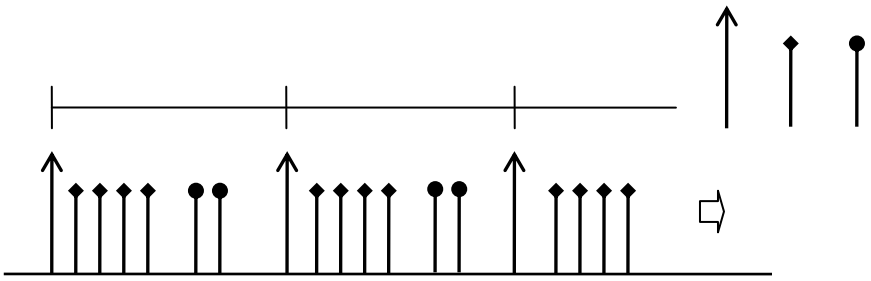
message: 601(ID) 2B 40 60 00 IF 00 00 00 // Set 6040h as1F to IP_ACTIVE status;

Activate the communicaiton
message: 00(ID) 01 01

Setting the second axis is the same with the first axis.

Host sends according to the setting of test SYNC cycle(1ms):
message: 201(ID) 10 00 00 00 //16 P/R, interpolation potion data of the first axis
message: 202(ID) 20 00 00 00 //32 P/R, interpolation potion data of the second axis
message: 80(ID) // cycle to send SYNC according to interpolation cycle
And then,drive returns
message: 181(ID) xx xx 00 00 //status word of the first axis
message: 182(ID) xx xx xx xx //position and speed of the first axis
message: 281(ID) xx xx 00 00 //status word of the second axis
message: 282(ID) xx xx xx xx //position and speed of the second axis

Sequence diagram:



9.5 Homing

Set the control mode as homing control.
message: 601(ID) 2F 60 60 00 06 00 00 00// Set the control mode as homing control.

message: 601(ID) 2F 98 60 00 04 00 00 00//Use the fourth way to set the homing mode.

Set the status machine
message: 601(ID) 2B 40 60 00 06 00 00 00
message: 601(ID) 2B 40 60 00 07 00 00 00
message: 601(ID) 2B 40 60 00 0F 00 00 00 //Servo On

Send data through PDO. (Set PDO1 as 6040(status word). Set PDO2 as 607A(Position pulse number) and 6081. (Speed, unit 0.1rpm)

Set the homing method as 10rpm.
message：601(ID) 23 99 60 02 64 00 00 00

Homing is started.
message：201 (ID) 1F 00

Cancel homing.
message：201 (ID) 0F 00

Chapter 10 Other function

10.1 Digital inputs and outputs

60FE (Physical outputs)

In some case, the switch (homing and limit switch) is committed by upper computer not servo drive.the signal is committed by object 60FE-01h(Physical outputs).

Index	60FE $_{h}$
Name	Digital outputs
Object Code	ARRAY
No. of Elements	2
Data Type	UINT32

Sub-Index	01 $_{h}$
Name	Physical outputs
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Default Value	0

Sub-Index	02 $_{h}$
Name	Bit mask
Object Code	VAR
Data Type	UINT32
Access	RW
PDO Mapping	YES
Default Value	0

Bit0-15	Bit16	Bit17	Bit18	Bit19	Bit20
reserved	CN1_in1	CN1_in2	CN1_in3	CN1_in4	CN1_in5
Bit21	Bit22	Bit23	Bit24	Bit25	Bit25-31
CN1_in6	CN1_in7	CN1_in8	CN1_out1	CN1_out2	reserved

Bit16-bit23 of the object corresponding input signal terminal connector(CN1). The function of the terminals is formed by Pn509/510 or inversing Pn516/517. Enabled bus commit by Pn512/Pn513 for the bus commit bits. Refer to <ProNet Series AC Servo User's Manual>. Bit24-bit25 of the object corresponding input signal terminal connector(CN1). The function of the terminals is formed by Pn511 or inversing Pn528. Enabled bus commit by Pn512/Pn513 for the bus commit bits. Refer to <ProNet Plus Series AC Servo User's Manual>.

60FD (Physical iutputs)

The host can read the object 60FDh(Digital Inputs) to monitor the switching inputs of the servo.

Index	60FD $_{h}$
Name	Digital iutputs
Object Code	Variable
Data Type	UINT32

Sub-Index	00 $_{h}$
Name	Physical iutputs
Object Code	VAR
Data Type	UINT32
Access	RO
PDO Mapping	YES
Default Value	0

Bit0	Bit1	Bit2	Bit3-15	Bit16	Bit17	Bit18
negative limit switch	positive limit switch	home switch	reserved	CN1_in1	CN1_in2	CN1_in3
Bit19	Bit20	Bit21	Bit22	Bit23	Bit24-31	
CN1_in4	CN1_in5	CN1_in6	CN1_in7	CN1_in8	reserved	

The terminals of Pn509、Pn510、Pn511 refer to <ProNet Plus Series AC Servo User's Manual>.

10.2 Dummy object

To a better performance and save bandwidth of the bus ,the host can post data to the different slaves by the same PDO. COB-ID of slave's RPDO should be set as the same value,and dummy object should be mapped to slave's RPDO. Sometimes need to turn off some needless TPDO.

Example of mapping the first RPDO of two nodes:

Node1

0x1600-1 = 60C1 00 10H // Interpolated position

0x1600-2 = 0007 00 10H // Dummy object 32 bit

0x1600-0 = 2H

Node 2

0x1600-1 = 0007 00 10H // Dummy object 32 bit

0x1600-2 = 60C1 00 10H // Interpolated position

0x1600-0 = 2H

Host controller can use one PDO，and map the two slaves'interpolated position to the first and the second objects at the same time.

Example of programming COB-ID: revise COB-ID of Node 2 the first RPDO from 0x202 to 0x201

0x1400-1 = 80000201H //Restrict PDO，and write new COB-ID

0x1400-1 = 00000201H // Enabled PDO

Example of closing TPDO: restrict the second TPDO of Node 2 from transmitting （default COB-ID is 0x282）

0x1801-1 = 80000282H

Appendix Object dictionary

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
2	--	VAR	od_integer8	INT8	RW	YES	●						
3	--	VAR	od_integer16	INT16	RW	YES	●						
4	--	VAR	od_integer32	INT32	RW	YES	●						
5	--	VAR	od_unsigned8	UINT8	RW	YES	●						
6	--	VAR	od_unsigned16	UINT16	RW	YES	●						
7	--	VAR	od_unsigned32	UINT32	RW	YES	●						
1000	--	VAR	device_type	UINT32	RO	NO	●						
1001	--	VAR	error_register	UINT8	RO	NO	●						
1003	--	VAR	pre_defined_error_field	UINT8	RW	NO	●						
1005	--	VAR	cob_id_sync	UINT32	RW	NO	●						
1006	--	VAR	communication_cycle_period	UINT32	RW	NO	●						
1007	--	VAR	synchronous_window_length	UINT32	RW	NO	●						
1014	--	VAR	cob_id_emergency_message	UINT32	RW	NO	●						
1016	--	ARRAY	consumer_heartbeat_time	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		consumer_heartbeat_time1	UINT32	RW	NO	●						
1017		VAR	producer_heartbeat_time	UINT16	RW	NO	●						
1018	--	RECORD	identity_object	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
	1		vendor_id	UINT32	RO	NO	●						
	2		product_code	UINT32	RO	NO	●						
	3		revision_number	UINT32	RO	NO	●						
	4		serial_number	UINT32	RO	NO	●						
1029	--	ARRAY	error_behaviour	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		communication_error	UINT8	RW	NO	●						
1200	--	RECORD	server_sdo_parameter	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		cob_id_client_server	UINT32	RO	NO	●						
	2		cob_id_server_client	UINT32	RO	NO	●						
1400	--	RECORD	receive_pdo_parameter_rpdo1	--	--	--	●						
	0		number_of_entries_rpdo1	UINT8	RO	NO	●						
	1		cob_id_used_by_pdo_rpdo1	UINT32	RO	NO	●						
	2		transmission_type_rpdo1	UINT8	RW	NO	●						
1401	--	RECORD	receive_pdo_parameter_rpdo2	--	--	--	●						
	0		number_of_entries_rpdo2	UINT8	RO	NO	●						
	1		cob_id_used_by_pdo_rpdo2	UINT32	RO	NO	●						
	2		transmission_type_rpdo2	UINT8	RW	NO	●						
1402	--	RECORD	receive_pdo_parameter_rpdo3	--	--	--	●						
	0		number_of_entries_rpdo3	UINT8	RO	NO	●						
	1		cob_id_used_by_pdo_rpdo3	UINT32	RO	NO	●						
	2		transmission_type_rpdo3	UINT8	RW	NO	●						
1403	--	RECORD	receive_pdo_parameter_rpdo4	--	--	--	●						
	0		number_of_entries_rpdo4	UINT8	RO	NO	●						

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
	1		cob_id_used_by_pdo_rpdo4	UINT32	RO	NO	●						
	2		transmission_type_rpdo4	UINT8	RW	NO	●						
1600	--	RECORD	receive_pdo_mapping_rpdo1	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		first_mapped_object_rpdo1	UINT32	RW	NO	●						
	2		second_mapped_object_rpdo1	UINT32	RW	NO	●						
	3		third_mapped_object_rpdo1	UINT32	RW	NO	●						
	4		fourth_mapped_object_rpdo1	UINT32	RW	NO	●						

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
1601	--	RECORD	receive_pdo_mapping_rpdo2	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		first_mapped_object_rpdo2	UINT32	RW	NO	●						
	2		second_mapped_object_rpdo2	UINT32	RW	NO	●						
	3		third_mapped_object_rpdo2	UINT32	RW	NO	●						
	4		fourth_mapped_object_rpdo2	UINT32	RW	NO	●						
1602	--	RECORD	receive_pdo_mapping_rpdo3	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		first_mapped_object_rpdo3	UINT32	RW	NO	●						
	2		second_mapped_object_rpdo3	UINT32	RW	NO	●						
	3		third_mapped_object_rpdo3	UINT32	RW	NO	●						
	4		fourth_mapped_object_rpdo3	UINT32	RW	NO	●						
1603	--	RECORD	receive_pdo_mapping_rpdo4	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		first_mapped_object_rpdo4	UINT32	RW	NO	●						
	2		second_mapped_object_rpdo4	UINT32	RW	NO	●						
	3		third_mapped_object_rpdo4	UINT32	RW	NO	●						
	4		fourth_mapped_object_rpdo4	UINT32	RW	NO	●						
1800	--	RECORD	transmit_pdo_parameter_tpdo1	--	--	--	●						
	0		number_of_entries_tpdo1	UINT32	RO	NO	●						
	1		cob_id_used_by_pdo_tpdo1	UINT32	RO	NO	●						
	2		transmission_type_tpdo1	UINT8	RW	NO	●						
	3		inhibit_time_tpdo1	UINT16	RW	NO	●						
	5		event_timer_tpdo1	UINT16	RW	NO	●						

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
1801	--	RECORD	transmit_pdo_parameter_tpdo2	--	--	--	●						
	0		number_of_entries_tpdo2	UINT32	RO	NO	●						
	1		cob_id_used_by_pdo_tpdo2	UINT32	RO	NO	●						
	2		transmission_type_tpdo2	UINT8	RW	NO	●						
	3		inhibit_time_tpdo2	UINT16	RW	NO	●						
	5		event_timer_tpdo2	UINT16	RW	NO	●						
1802	--	RECORD	transmit_pdo_parameter_tpdo3	--	--	--	●						
	0		number_of_entries_tpdo3	UINT32	RO	NO	●						
	1		cob_id_used_by_pdo_tpdo3	UINT32	RO	NO	●						
	2		transmission_type_tpdo3	UINT8	RW	NO	●						
	3		inhibit_time_tpdo3	UINT16	RW	NO	●						
	5		event_timer_tpdo3	UINT16	RW	NO	●						
1803	--	RECORD	transmit_pdo_parameter_tpdo4	--	--	--	●						
	0		number_of_entries_tpdo4	UINT32	RO	NO	●						
	1		cob_id_used_by_pdo_tpdo4	UINT32	RO	NO	●						
	2		transmission_type_tpdo4	UINT8	RW	NO	●						
	3		inhibit_time_tpdo4	UINT16	RW	NO	●						
	5		event_timer_tpdo4	UINT16	RW	NO	●						
1A00	--	RECORD	transmit_pdo_mapping_tpdo1	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		first_mapped_object_tpdo1	UINT32	RW	NO	●						
	2		second_mapped_object_tpdo1	UINT32	RW	NO	●						
	3		third_mapped_object_tpdo1	UINT32	RW	NO	●						
	4		fourth_mapped_object_tpdo1	UINT32	RW	NO	●						

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
1A01	--	RECORD	transmit_pdo_mapping_tpdo2	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		first_mapped_object_tpdo2	UINT32	RW	NO	●						
	2		second_mapped_object_tpdo2	UINT32	RW	NO	●						
	3		third_mapped_object_tpdo2	UINT32	RW	NO	●						
	4		fourth_mapped_object_tpdo2	UINT32	RW	NO	●						
1A02	--	RECORD	transmit_pdo_mapping_tpdo3	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		first_mapped_object_tpdo3	UINT32	RW	NO	●						
	2		second_mapped_object_tpdo3	UINT32	RW	NO	●						
	3		third_mapped_object_tpdo3	UINT32	RW	NO	●						
	4		fourth_mapped_object_tpdo3	UINT32	RW	NO	●						
1A03	--	RECORD	transmit_pdo_mapping_tpdo4	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		first_mapped_object_tpdo4	UINT32	RW	NO	●						
	2		second_mapped_object_tpdo4	UINT32	RW	NO	●						
	3		third_mapped_object_tpdo4	UINT32	RW	NO	●						
	4		fourth_mapped_object_tpdo4	UINT32	RW	NO	●						

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
2000	--	RECORD	mask_tpdo1	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		mask1_tpdo1	UINT32	RW	NO	●						
	2		mask2_tpdo1	UINT32	RW	NO	●						
2001	--	RECORD	mask_tpdo2	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		mask1_tpdo2	UINT32	RW	NO	●						
	2		mask2_tpdo2	UINT32	RW	NO	●						
2002	--	RECORD	mask_tpdo3	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		mask1_tpdo3	UINT32	RW	NO	●						
	2		mask2_tpdo3	UINT32	RW	NO	●						
2003	--	RECORD	mask_tpdo4	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		mask1_tpdo4	UINT32	RW	NO	●						
	2		mask2_tpdo4	UINT32	RW	NO	●						
2105	0	VAR	sync_time_period	UINT32	RW	NO	●						

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
3000	--	VAR	Pn000_a	UINT16	RW	NO	●						--
3001	--	VAR	Pn001_a	UINT16	RW	NO	●						--
3002	--	VAR	Pn002_a	UINT16	RW	NO	●						--
3003	--	VAR	Pn003_a	UINT16	RW	NO	●						--
3004	--	VAR	Pn004_a	UINT16	RW	NO	●						--
3005	--	VAR	Pn005_a	UINT16	RW	NO	●						--
3006	--	VAR	Pn006_a	UINT16	RW	NO	●						--
3007	--	VAR	Pn007_a	UINT16	RW	NO	●						--
3008	--	VAR	Pn008_a	UINT16	RW	NO	●						--
3009	--	VAR	Pn009_a	UINT16	RW	NO	●						--
300A	--	VAR	Pn010_a	UINT16	RW	NO	●						--
3010	--	VAR	Pn100_a	UINT16	RW	NO	●						--
3011	--	VAR	Pn101_a	UINT16	RW	NO	●						--
3012	--	VAR	Pn102_a	UINT16	RW	NO	●						Hz
3013	--	VAR	Pn103_a	UINT16	RW	NO	●						0.1ms
3014	--	VAR	Pn104_a	UINT16	RW	NO	●						1/s
3015	--	VAR	Pn105_a	UINT16	RW	NO	●						0.1ms
3016	--	VAR	Pn106_a	UINT16	RW	NO	●						--
3017	--	VAR	Pn107_a	UINT16	RW	NO	●						Hz
3018	--	VAR	Pn108_a	UINT16	RW	NO	●						0.1ms
3019	--	VAR	Pn109_a	UINT16	RW	NO	●						Hz
301A	--	VAR	Pn110_a	UINT16	RW	NO	●						0.1ms
301B	--	VAR	Pn111_a	UINT16	RW	NO	●						r/min
301C	--	VAR	Pn112_a	UINT16	RW	NO	●						%

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
301D	--	VAR	Pn113_a	UINT16	RW	NO	●						0.1ms
301E	--	VAR	Pn114_a	UINT16	RW	NO	●						%
301F	--	VAR	Pn115_a	UINT16	RW	NO	●						0.1ms
3020	--	VAR	Pn116_a	UINT16	RW	NO	●						--
3021	--	VAR	Pn117_a	UINT16	RW	NO	●						%
3022	--	VAR	Pn118_a	UINT16	RW	NO	●						--
3023	--	VAR	Pn119_a	UINT16	RW	NO	●						--
3024	--	VAR	Pn120_a	UINT16	RW	NO	●						--
3025	--	VAR	Pn121_a	UINT16	RW	NO	●						--
3026	--	VAR	Pn122_a	UINT16	RW	NO	●						0.1ms
3027	--	VAR	Pn123_a	UINT16	RW	NO	●						--
3028	--	VAR	Pn124_a	UINT16	RW	NO	●						--
3029	--	VAR	Pn125_a	UINT16	RW	NO	●						0.1ms
302A	--	VAR	Pn126_a	UINT16	RW	NO	●						--
302B	--	VAR	Pn127_a	UINT16	RW	NO	●						0.1ms
302C	--	VAR	Pn128_a	UINT16	RW	NO	●						0.1%
302D	--	VAR	Pn129_a	UINT16	RW	NO	●						r/min
302E	--	VAR	Pn130_a	UINT16	RW	NO	●						0.1%
302F	--	VAR	Pn131_a	UINT16	RW	NO	●						r/min
3030	--	VAR	Pn132_a	UINT16	RW	NO	●						0.1%/1000rpm
3042	--	VAR	Pn200_a	UINT16	RW	NO	●						--
3043	--	VAR	Pn201_a	UINT16	RW	NO	●						--
3044	--	VAR	Pn202_a	UINT16	RW	NO	●						--
3045	--	VAR	Pn203_a	UINT16	RW	NO	●						--
3046	--	VAR	Pn204_a	UINT16	RW	NO	●						0.1ms

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
3047	--	VAR	Pn205_a	UINT16	RW	NO	●						--
3049	--	VAR	Pn207_a	UINT16	RW	NO	●						--
305A	--	VAR	Pn208_a	UINT16	RW	NO	●						--
3050	--	VAR	Pn300_a	UINT16	RW	NO	●						rmp/V
3051	--	VAR	Pn301_a	UINT16	RW	NO	●						10mv
3054	--	VAR	Pn304_a	UINT16	RW	NO	●						rmp
3055	--	VAR	Pn305_a	UINT16	RW	NO							rpm
3056	--	VAR	Pn306_a	UINT16	RW	NO	●						ms
3057	--	VAR	Pn307_a	UINT16	RW	NO	●						ms
3058	--	VAR	Pn308_a	UINT16	RW	NO	●						ms
3059	--	VAR	Pn309_a	UINT16	RW	NO	●						ms
305A	--	VAR	Pn310_a	UINT16	RW	NO	●						--
305B	--	VAR	Pn311_a	UINT16	RW	NO	●						--
305C	--	VAR	Pn312_a	UINT16	RW	NO	●						rmp
3060	--	VAR	Pn316_a	UINT16	RW	NO	●						--
3061	--	VAR	Pn317_a	UINT16	RW	NO	●						--
3062	--	VAR	Pn318_a	UINT16	RW	NO	●						--
3063	--	VAR	Pn319_a	UINT16	RW	NO	●						--
3064	--	VAR	Pn320_a	UINT16	RW	NO	●						--
3065	--	VAR	Pn321_a	UINT16	RW	NO	●						--
3066	--	VAR	Pn322_a	UINT16	RW	NO	●						--
3067	--	VAR	Pn400_a	UINT16	RW	NO	●						0.1v/100%
3068	--	VAR	Pn401_a	UINT16	RW	NO	●						%
3069	--	VAR	Pn402_a	UINT16	RW	NO	●						%
306A	--	VAR	Pn403_a	UINT16	RW	NO	●						%

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
306B	--	VAR	Pn404_a	UINT16	RW	NO	●						%
306C	--	VAR	Pn405_a	UINT16	RW	NO	●						%
306D	--	VAR	Pn406_a	UINT16	RW	NO	●						rpm
306E	--	VAR	Pn407_a	UINT16	RW	NO	●						Hz
306F	--	VAR	Pn408_a	UINT16	RW	NO	●						--
3070	--	VAR	Pn409_a	UINT16	RW	NO	●						Hz
3071	--	VAR	Pn410_a	UINT16	RW	NO	●						--
3072	--	VAR	Pn411_a	UINT16	RW	NO	●						0.1 Hz
3073	--	VAR	Pn412_a	UINT16	RW	NO	●						--
3074	--	VAR	Pn413_a	UINT16	RW	NO	●						0.1ms
3075	--	VAR	Pn414_a	UINT16	RW	NO	●						rpm
3076	--	VAR	Pn415_a	UINT16	RW	NO	●						10mv
3078	--	VAR	Pn500_a	UINT16	RW	NO	●						pulse
3079	--	VAR	Pn501_a	UINT16	RW	NO	●						rpm
307A	--	VAR	Pn502_a	UINT16	RW	NO	●						rpm
307B	--	VAR	Pn503_a	UINT16	RW	NO	●						rpm
307C	--	VAR	Pn504_a	UINT16	RW	NO	●						256pulse
307D	--	VAR	Pn505_a	INT16	RW	NO	●						ms
307E	--	VAR	Pn506_a	UINT16	RW	NO	●						10ms
307F	--	VAR	Pn507_a	UINT16	RW	NO	●						rpm
3080	--	VAR	Pn508_a	UINT16	RW	NO	●						10ms
3081	--	VAR	Pn509_a	UINT16	RW	NO	●						--
3082	--	VAR	Pn510_a	UINT16	RW	NO	●						--
3083	--	VAR	Pn511_a	UINT16	RW	NO	●						--
3084	--	VAR	Pn512_a	UINT16	RW	NO	●						--

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
3085	--	VAR	Pn513_a	UINT16	RW	NO	●						--
3086	--	VAR	Pn514_a	UINT16	RW	NO	●						0.2ms
3088	--	VAR	Pn516_a	UINT16	RW	NO	●						--
3089	--	VAR	Pn517_a	UINT16	RW	NO	●						--
308A	--	VAR	Pn518_a	UINT16	RW	NO	●						ms
308B	--	VAR	Pn519_a	UINT16	RW	NO	●						0.1ms
308C	--	VAR	Pn520_a	UINT16	RW	NO	●						0.1ms
308D	--	VAR	Pn521_a	UINT16	RW	NO	●						--
3091	--	VAR	Pn525_a	UINT16	RW	NO	●						%
3092	--	VAR	Pn526_a	UINT16	RW	NO	●						℃
3094	--	VAR	Pn528_a	UINT16	RW	NO	●						--
3095	--	VAR	Pn529_a	UINT16	RW	NO	●						%
3096	--	VAR	Pn530_a	UINT16	RW	NO	●						ms
3098	--	VAR	Pn600_a	UINT16	RW	NO	●						--
3099	--	VAR	Pn601_a	UINT16	RW	NO	●						--
30B8	--	VAR	Pn632_a	UINT16	RW	NO	●						rmp
30C8	--	VAR	Pn648_a	UINT16	RW	NO	●						0.1ms
30D8	--	VAR	Pn664_a	UINT16	RW	NO	●						50ms
30E7	--	VAR	Pn679_a	UINT16	RW	NO	●						50ms
30E9	--	VAR	Pn681_a	UINT16	RW	NO	●						--
30EA	--	VAR	Pn682_a	UINT16	RW	NO	●						--
30EB	--	VAR	Pn683_a	UINT16	RW	NO	●						--
30EC	--	VAR	Pn684_a	UINT16	RW	NO	●						--
30ED	--	VAR	Pn685_a	UINT16	RW	NO	●						rmp
30EE	--	VAR	Pn686_a	UINT16	RW	NO	●						rmp

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
30EF	--	VAR	Pn687_a	UINT16	RW	NO	●						--
30F1	--	VAR	Pn689_a	UINT16	RW	NO	●						--
30F3	--	VAR	Pn691_a	UINT16	RW	NO	●						1p
30FC	--	VAR	Pn700_a	UINT16	RW	NO	●						--
30FD	--	VAR	Pn701_a	UINT16	RW	NO	●						--
30FF	--	VAR	Pn703_a	UINT16	RW	NO	●						--
3100	--	VAR	Pn704_a	UINT16	RW	NO	●						--
3101	--	VAR	Pn705_a	UINT16	RW	NO	●						--
3102	--	VAR	Pn706_a	UINT16	RW	NO	●						--
3103	--	VAR	Pn707_a	UINT16	RW	NO	●						--
3104	--	VAR	Pn708_a	UINT16	RW	NO	●						--
3105	--	VAR	Pn709_a	UINT16	RW	NO	●						--
3106	--	VAR	Pn710_a	UINT16	RW	NO	●						--
3138	--	VAR	Pn840_a	UINT16	RW	NO	●						--
3200	--	VAR	Fn000-0_a	UINT16	RO	NO	●						--
3201	--	VAR	Fn000-1_a	UINT16	RO	NO	●						--
3202	--	VAR	Fn000-2_a	UINT16	RO	NO	●						--
3203	--	VAR	Fn000-3_a	UINT16	RO	NO	●						--
3204	--	VAR	Fn000-4_a	UINT16	RO	NO	●						--
3205	--	VAR	Fn000-5_a	UINT16	RO	NO	●						--
3206	--	VAR	Fn000-6_a	UINT16	RO	NO	●						--
3207	--	VAR	Fn000-7_a	UINT16	RO	NO	●						--
3208	--	VAR	Fn000-8_a	UINT16	RO	NO	●						--
3209	--	VAR	Fn000-9_a	UINT16	RO	NO	●						--
320A	--	VAR	Un000_a	UINT16	RO	NO	●						r/min

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
320B	--	VAR	Un001_a	UINT16	RO	NO	●						r/min
320C	--	VAR	Un002_a	UINT16	RO	NO	●						%
320D	--	VAR	Un003_a	UINT16	RO	NO	●						%
320E	--	VAR	Un004_a	UINT16	RO	NO	●						pulse
320F	--	VAR	Un005_a	UINT16	RO	NO	●						--
3210	--	VAR	Un006_a	UINT16	RO	NO	●						--
3211	--	VAR	Un007_a	UINT16	RO	NO	●						--
3212	--	VAR	Un008_a	UINT16	RO	NO	●						1kHz
3213	--	VAR	Un009_a	UINT16	RO	NO	●						pulse
3214	--	VAR	Un010_a	UINT16	RO	NO	●						--
3215	--	VAR	Un011_a	UINT16	RO	NO	●						--
3216	--	VAR	Un012_a	UINT16	RO	NO	●						--
3217	--	VAR	Un013_a	UINT16	RO	NO	●						--
3218	--	VAR	Un014_a	UINT16	RO	NO	●						--
*3219	--	VAR	Un015_a	UINT16	RO	NO	●						--
*321A	--	VAR	Un016_a	UINT16	RO	NO	●						--
321E	--	VAR	DSP-Edition_a	UINT16	RO	NO	●						--
321B	--	VAR	CO_Fn010	UINT16	RW	NO	●						--
321C	--	VAR	CO_Fn011	UINT16	RW	NO	●						--
*3226	--	VAR	CO_Fn010_AMG	UINT16	RW	NO	●						--
*3227	--	VAR	CO_Fn011_AMG	UINT16	RW	NO	●						--
3300	--	VAR	Rotates_a	UINT16	RO	NO	●						--
3301	--	VAR	singlePos_a	UINT16	RO	NO	●						--
Those objects as below are apply to EDS/ETS only.													
3800	--	VAR	Pn000_b	UINT16	RW	NO	●						--

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
3801	--	VAR	Pn001_b	UINT16	RW	NO	●						--
3802	--	VAR	Pn002_b	UINT16	RW	NO	●						--
3803	--	VAR	Pn003_b	UINT16	RW	NO	●						--
3804	--	VAR	Pn004_b	UINT16	RW	NO	●						--
3805	--	VAR	Pn005_b	UINT16	RW	NO	●						--
3806	--	VAR	Pn006_b	UINT16	RW	NO	●						--
3807	--	VAR	Pn007_b	UINT16	RW	NO	●						--
3810	--	VAR	Pn100_b	UINT16	RW	NO	●						--
3811	--	VAR	Pn101_b	UINT16	RW	NO	●						--
3812	--	VAR	Pn102_b	UINT16	RW	NO	●						Hz
3813	--	VAR	Pn103_b	UINT16	RW	NO	●						0.1ms
3814	--	VAR	Pn104_b	UINT16	RW	NO	●						1/s
3815	--	VAR	Pn105_b	UINT16	RW	NO	●						0.1ms
3816	--	VAR	Pn106_b	UINT16	RW	NO	●						--
3817	--	VAR	Pn107_b	UINT16	RW	NO	●						Hz
3818	--	VAR	Pn108_b	UINT16	RW	NO	●						0.1ms
3819	--	VAR	Pn109_b	UINT16	RW	NO	●						Hz
381A	--	VAR	Pn110_b	UINT16	RW	NO	●						0.1ms
381B	--	VAR	Pn111_b	UINT16	RW	NO	●						r/min
381C	--	VAR	Pn112_b	UINT16	RW	NO	●						%
381D	--	VAR	Pn113_b	UINT16	RW	NO	●						0.1ms
381E	--	VAR	Pn114_b	UINT16	RW	NO	●						%
381F	--	VAR	Pn115_b	UINT16	RW	NO	●						0.1ms
3820	--	VAR	Pn116_b	UINT16	RW	NO	●						--
3821	--	VAR	Pn117_b	UINT16	RW	NO	●						%

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
3822	--	VAR	Pn118_b	UINT16	RW	NO	●						--
3823	--	VAR	Pn119_b	UINT16	RW	NO	●						--
3824	--	VAR	Pn120_b	UINT16	RW	NO	●						--
3825	--	VAR	Pn121_b	UINT16	RW	NO	●						--
3826	--	VAR	Pn122_b	UINT16	RW	NO	●						0.1ms
3827	--	VAR	Pn123_b	UINT16	RW	NO	●						--
3828	--	VAR	Pn124_b	UINT16	RW	NO	●						--
3829	--	VAR	Pn125_b	UINT16	RW	NO	●						0.1ms
382A	--	VAR	Pn126_b	UINT16	RW	NO	●						--
382B	--	VAR	Pn127_b	UINT16	RW	NO	●						0.1ms
382C	--	VAR	Pn128_b	UINT16	RW	NO	●						0.1%
382D	--	VAR	Pn129_b	UINT16	RW	NO	●						r/min
382E	--	VAR	Pn130_b	UINT16	RW	NO	●						0.1%
382F	--	VAR	Pn131_b	UINT16	RW	NO	●						r/min
3830	--	VAR	Pn132_b	UINT16	RW	NO	●						0.1%/1000rpm
3855	--	VAR	Pn305_b	UINT16	RW	NO	●						rpm
3856	--	VAR	Pn306_b	UINT16	RW	NO	●						ms
3857	--	VAR	Pn307_b	UINT16	RW	NO	●						ms
3858	--	VAR	Pn308_b	UINT16	RW	NO	●						ms
3859	--	VAR	Pn309_b	UINT16	RW	NO	●						ms
385A	--	VAR	Pn310_b	UINT16	RW	NO	●						--
385B	--	VAR	Pn311_b	UINT16	RW	NO	●						--
3868	--	VAR	Pn401_b	UINT16	RW	NO	●						%
3869	--	VAR	Pn402_b	UINT16	RW	NO	●						%
386A	--	VAR	Pn403_b	UINT16	RW	NO	●						%

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
386B	--	VAR	Pn404_b	UINT16	RW	NO	●						%
386C	--	VAR	Pn405_b	UINT16	RW	NO	●						%
386D	--	VAR	Pn406_b	UINT16	RW	NO	●						rpm
386E	--	VAR	Pn407_b	UINT16	RW	NO	●						Hz
386F	--	VAR	Pn408_b	UINT16	RW	NO	●						--
3870	--	VAR	Pn409_b	UINT16	RW	NO	●						Hz
3871	--	VAR	Pn410_b	UINT16	RW	NO	●						--
3872	--	VAR	Pn411_b	UINT16	RW	NO	●						0.1 Hz
3873	--	VAR	Pn412_b	UINT16	RW	NO	●						--
3874	--	VAR	Pn413_b	UINT16	RW	NO	●						0.1ms
3875	--	VAR	Pn414_b	UINT16	RW	NO	●						rpm
3878	--	VAR	Pn500_b	UINT16	RW	NO	●						pulse
3879	--	VAR	Pn501_b	UINT16	RW	NO	●						rpm
387A	--	VAR	Pn502_b	UINT16	RW	NO	●						rpm
387B	--	VAR	Pn503_b	UINT16	RW	NO	●						rpm
387C	--	VAR	Pn504_b	UINT16	RW	NO	●						256pulse
387D	--	VAR	Pn505_b	INT16	RW	NO	●						ms
387E	--	VAR	Pn506_b	UINT16	RW	NO	●						10ms
387F	--	VAR	Pn507_b	UINT16	RW	NO	●						rpm
3880	--	VAR	Pn508_b	UINT16	RW	NO	●						10ms
3881	--	VAR	Pn509_b	UINT16	RW	NO	●						--
3882	--	VAR	Pn510_b	UINT16	RW	NO	●						--
3883	--	VAR	Pn511_b	UINT16	RW	NO	●						--
3884	--	VAR	Pn512_b	UINT16	RW	NO	●						--
3885	--	VAR	Pn513_b	UINT16	RW	NO	●						--

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
3886	--	VAR	Pn514_b	UINT16	RW	NO	●						0.2ms
3888	--	VAR	Pn516_b	UINT16	RW	NO	●						--
3889	--	VAR	Pn517_b	UINT16	RW	NO	●						--
3891	--	VAR	Pn525_b	UINT16	RW	NO	●						%
38FC	--	VAR	Pn700_b	UINT16	RW	NO	●						--
38FD	--	VAR	Pn701_b	UINT16	RW	NO	●						--
38FF	--	VAR	Pn703_b	UINT16	RW	NO	●						--
3900	--	VAR	Pn704_b	UINT16	RW	NO	●						--
3A00	--	VAR	Fn000-0_b	UINT16	RO	NO	●						--
3A01	--	VAR	Fn000-1_b	UINT16	RO	NO	●						--
3A02	--	VAR	Fn000-2_b	UINT16	RO	NO	●						--
3A03	--	VAR	Fn000-3_b	UINT16	RO	NO	●						--
3A04	--	VAR	Fn000-4_b	UINT16	RO	NO	●						--
3A05	--	VAR	Fn000-5_b	UINT16	RO	NO	●						--
3A06	--	VAR	Fn000-6_b	UINT16	RO	NO	●						--
3A07	--	VAR	Fn000-7_b	UINT16	RO	NO	●						--
3A08	--	VAR	Fn000-8_b	UINT16	RO	NO	●						--
3A09	--	VAR	Fn000-9_b	UINT16	RO	NO	●						--
3A0A	--	VAR	Un000_b	UINT16	RO	NO	●						r/min
3A0B	--	VAR	Un001_b	UINT16	RO	NO	●						r/min
3A0C	--	VAR	Un002_b	UINT16	RO	NO	●						%
3A0D	--	VAR	Un003_b	UINT16	RO	NO	●						%
3A0E	--	VAR	Un004_b	UINT16	RO	NO	●						pulse
3A0F	--	VAR	Un005_b	UINT16	RO	NO	●						--
3A10	--	VAR	Un006_b	UINT16	RO	NO	●						--

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
3A11	--	VAR	Un007_b	UINT16	RO	NO	●						--
3A12	--	VAR	Un008_b	UINT16	RO	NO	●						1kHz
3A13	--	VAR	Un009_b	UINT16	RO	NO	●						pulse
3A14	--	VAR	Un010_b	UINT16	RO	NO	●						--
3A15	--	VAR	Un011_b	UINT16	RO	NO	●						--
3A16	--	VAR	Un012_b	UINT16	RO	NO	●						--
3A17	--	VAR	Un013_b	UINT16	RO	NO	●						--
3A18	--	VAR	Un014_b	UINT16	RO	NO	●						--
3A1B	--	VAR	CO_Fn010_b	UINT16	RW	NO	●						--
3A1C	--	VAR	CO_Fn011_b	UINT16	RW	NO	●						--
3A1E	--	VAR	DSP-Edition_b	UINT16	RO	NO	●						--
3B00	--	VAR	Rotates_b	UINT16	RO	NO	●						--
3B01	--	VAR	singlePos_b	UINT16	RO	NO	●						--
Those objects as below are applied to ETS only.													
4000	--	VAR	Pn000_c	UINT16	RW	NO	●						--
4001	--	VAR	Pn001_c	UINT16	RW	NO	●						--
4002	--	VAR	Pn002_c	UINT16	RW	NO	●						--
4003	--	VAR	Pn003_c	UINT16	RW	NO	●						--
4004	--	VAR	Pn004_c	UINT16	RW	NO	●						--
4005	--	VAR	Pn005_c	UINT16	RW	NO	●						--
4006	--	VAR	Pn006_c	UINT16	RW	NO	●						--
4007	--	VAR	Pn007_c	UINT16	RW	NO	●						--
4010	--	VAR	Pn100_c	UINT16	RW	NO	●						--
4011	--	VAR	Pn101_c	UINT16	RW	NO	●						--
4012	--	VAR	Pn102_c	UINT16	RW	NO	●						Hz

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
4013	--	VAR	Pn103_c	UINT16	RW	NO	●						0.1ms
4014	--	VAR	Pn104_c	UINT16	RW	NO	●						1/s
4015	--	VAR	Pn105_c	UINT16	RW	NO	●						0.1ms
4016	--	VAR	Pn106_c	UINT16	RW	NO	●						--
4017	--	VAR	Pn107_c	UINT16	RW	NO	●						Hz
4018	--	VAR	Pn108_c	UINT16	RW	NO	●						0.1ms
4019	--	VAR	Pn109_c	UINT16	RW	NO	●						Hz
401A	--	VAR	Pn110_c	UINT16	RW	NO	●						0.1ms
401B	--	VAR	Pn111_c	UINT16	RW	NO	●						r/min
401C	--	VAR	Pn112_c	UINT16	RW	NO	●						%
401D	--	VAR	Pn113_c	UINT16	RW	NO	●						0.1ms
401E	--	VAR	Pn114_c	UINT16	RW	NO	●						%
401F	--	VAR	Pn115_c	UINT16	RW	NO	●						0.1ms
4020	--	VAR	Pn116_c	UINT16	RW	NO	●						--
4021	--	VAR	Pn117_c	UINT16	RW	NO	●						%
4022	--	VAR	Pn118_c	UINT16	RW	NO	●						--
4023	--	VAR	Pn119_c	UINT16	RW	NO	●						--
4024	--	VAR	Pn120_c	UINT16	RW	NO	●						--
4025	--	VAR	Pn121_c	UINT16	RW	NO	●						--
4026	--	VAR	Pn122_c	UINT16	RW	NO	●						0.1ms
4027	--	VAR	Pn123_c	UINT16	RW	NO	●						--
4028	--	VAR	Pn124_c	UINT16	RW	NO	●						--
4029	--	VAR	Pn125_c	UINT16	RW	NO	●						0.1ms
402A	--	VAR	Pn126_c	UINT16	RW	NO	●						--
402B	--	VAR	Pn127_c	UINT16	RW	NO	●						0.1ms

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
402C	--	VAR	Pn128_c	UINT16	RW	NO	●						0.1%
402D	--	VAR	Pn129_c	UINT16	RW	NO	●						r/min
402E	--	VAR	Pn130_c	UINT16	RW	NO	●						0.1%
402F	--	VAR	Pn131_c	UINT16	RW	NO	●						r/min
4030	--	VAR	Pn132_c	UINT16	RW	NO	●						0.1%/1000rpm
4055	--	VAR	Pn305_c	UINT16	RW	NO	●						rpm
4056	--	VAR	Pn306_c	UINT16	RW	NO	●						ms
4057	--	VAR	Pn307_c	UINT16	RW	NO	●						ms
4058	--	VAR	Pn308_c	UINT16	RW	NO	●						ms
4059	--	VAR	Pn309_c	UINT16	RW	NO	●						ms
405A	--	VAR	Pn310_c	UINT16	RW	NO	●						--
405B	--	VAR	Pn311_c	UINT16	RW	NO	●						--
4068	--	VAR	Pn401_c	UINT16	RW	NO	●						%
4069	--	VAR	Pn402_c	UINT16	RW	NO	●						%
406A	--	VAR	Pn403_c	UINT16	RW	NO	●						%
406B	--	VAR	Pn404_c	UINT16	RW	NO	●						%
406C	--	VAR	Pn405_c	UINT16	RW	NO	●						%
406D	--	VAR	Pn406_c	UINT16	RW	NO	●						rpm
406E	--	VAR	Pn407_c	UINT16	RW	NO	●						Hz
406F	--	VAR	Pn408_c	UINT16	RW	NO	●						--
4070	--	VAR	Pn409_c	UINT16	RW	NO	●						Hz
4071	--	VAR	Pn410_c	UINT16	RW	NO	●						--
4072	--	VAR	Pn411_c	UINT16	RW	NO	●						0.1 Hz
4073	--	VAR	Pn412_c	UINT16	RW	NO	●						--
4074	--	VAR	Pn413_c	UINT16	RW	NO	●						0.1ms

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
4075	--	VAR	Pn414_c	UINT16	RW	NO	●						rpm
4078	--	VAR	Pn500_c	UINT16	RW	NO	●						pulse
4079	--	VAR	Pn501_c	UINT16	RW	NO	●						rpm
407A	--	VAR	Pn502_c	UINT16	RW	NO	●						rpm
407B	--	VAR	Pn503_c	UINT16	RW	NO	●						rpm
407C	--	VAR	Pn504_c	UINT16	RW	NO	●						256pulse
407D	--	VAR	Pn505_c	INT16	RW	NO	●						ms
407E	--	VAR	Pn506_c	UINT16	RW	NO	●						10ms
407F	--	VAR	Pn507_c	UINT16	RW	NO	●						rpm
4080	--	VAR	Pn508_c	UINT16	RW	NO	●						10ms
4081	--	VAR	Pn509_c	UINT16	RW	NO	●						--
4082	--	VAR	Pn510_c	UINT16	RW	NO	●						--
4083	--	VAR	Pn511_c	UINT16	RW	NO	●						--
4084	--	VAR	Pn512_c	UINT16	RW	NO	●						--
4085	--	VAR	Pn513_c	UINT16	RW	NO	●						--
4086	--	VAR	Pn514_c	UINT16	RW	NO	●						0.2ms
4088	--	VAR	Pn516_c	UINT16	RW	NO	●						--
4089	--	VAR	Pn517_c	UINT16	RW	NO	●						--
4091	--	VAR	Pn525_c	UINT16	RW	NO	●						%
40FC	--	VAR	Pn700_c	UINT16	RW	NO	●						--
40FD	--	VAR	Pn701_c	UINT16	RW	NO	●						--
40FF	--	VAR	Pn703_c	UINT16	RW	NO	●						--
4100	--	VAR	Pn704_c	UINT16	RW	NO	●						--
4200	--	VAR	Fn000-0_c	UINT16	RO	NO	●						--
4201	--	VAR	Fn000-1_c	UINT16	RO	NO	●						--

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
4202	--	VAR	Fn000-2_c	UINT16	RO	NO	●						--
4203	--	VAR	Fn000-3_c	UINT16	RO	NO	●						--
4204	--	VAR	Fn000-4_c	UINT16	RO	NO	●						--
4205	--	VAR	Fn000-5_c	UINT16	RO	NO	●						--
4206	--	VAR	Fn000-6_c	UINT16	RO	NO	●						--
4207	--	VAR	Fn000-7_c	UINT16	RO	NO	●						--
4208	--	VAR	Fn000-8_c	UINT16	RO	NO	●						--
4209	--	VAR	Fn000-9_c	UINT16	RO	NO	●						--
420A	--	VAR	Un000_c	UINT16	RO	NO	●						r/min
420B	--	VAR	Un001_c	UINT16	RO	NO	●						r/min
420C	--	VAR	Un002_c	UINT16	RO	NO	●						%
420D	--	VAR	Un003_c	UINT16	RO	NO	●						%
420E	--	VAR	Un004_c	UINT16	RO	NO	●						pulse
420F	--	VAR	Un005_c	UINT16	RO	NO	●						--
4210	--	VAR	Un006_c	UINT16	RO	NO	●						--
4211	--	VAR	Un007_c	UINT16	RO	NO	●						--
4212	--	VAR	Un008_c	UINT16	RO	NO	●						1kHz
4213	--	VAR	Un009_c	UINT16	RO	NO	●						pulse
4214	--	VAR	Un010_c	UINT16	RO	NO	●						--
4215	--	VAR	Un011_c	UINT16	RO	NO	●						--
4216	--	VAR	Un012_c	UINT16	RO	NO	●						--
4217	--	VAR	Un013_c	UINT16	RO	NO	●						--
4218	--	VAR	Un014_a	UINT16	RO	NO	●						--
421B	--	VAR	CO_Fn010	UINT16	RW	NO	●						--
421C	--	VAR	CO_Fn011	UINT16	RW	NO	●						--

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
421E	--	VAR	DSP-Edition_c	UINT16	RO	NO	●						--
4300	--	VAR	Rotates_c	UINT16	RO	NO	●						--
4301	--	VAR	singlePosc_c	UINT16	RO	NO	●						--

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
603F	--	VAR	error_code_a	UINT16	RO	YES	●						--
6040	--	VAR	controlword_a	UINT16	RW	YES	●						--
6041	--	VAR	statusword_a	UINT16	RO	YES	●						--
605A	--	VAR	quick_stop_option_code_a	INT16	RW	NO	●						--
605B	--	VAR	shutdown_option_code_a	INT16	RW	NO	●						--
605C	--	VAR	disable_operation_option_code_a	INT16	RW	NO	●						--
605D	--	VAR	stop_option_code_a	INT16	RW	NO	●						--
605E	--	VAR	fault_reaction_option_code_a	INT16	RW	NO	●						--
6060	--	VAR	modes_of_operation_a	INT8	RW	YES	●						--
6061	--	VAR	modes_of_operation_display_a	INT8	RO	YES	●						--
6062	--	VAR	position_demand_value_a	INT32	RO	YES			●				position units
6063	--	VAR	position_actual_value*_a	INT32	RO	YES			●				inc
6064	--	VAR	position_actual_value_a	INT32	RO	YES		●	●		●	●	position units
6065	--	VAR	following_error_window_a	UINT32	RW	YES			●				position units
6066	--	VAR	following_error_time_out_a	UINT16	RW	YES			●				ms
6067	--	VAR	position_window_a	UINT32	RW	YES			●				position units
6068	--	VAR	position_window_time_a	UINT16	RW	YES			●				ms
6069	--	VAR	velocity_sensor_actual_value_a	UINT32	RO	YES				●			speed units
606B	--	VAR	velocity_demand_value_a	INT32	RO	YES				●			speed units
606C	--	VAR	velocity_actual_value_a	INT32	RO	YES	●						speed units
606D	--	VAR	velocity_window_a	UINT16	RW	YES				●			speed units
606E	--	VAR	velocity_window_time_a	UINT16	RW	YES				●			ms
606F	--	VAR	velocity_threshold_a	UINT16	RW	YES				●			speed units
6070	--	VAR	velocity_threshold_time_a	UINT16	RW	YES				●			ms
6071	--	VAR	target_torque_a	INT16	RW	YES					●		0.1% Tn

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
6072	--	VAR	Max_torque_a	UINT16	RW	YES					●		0.1% Tn
6074	--	VAR	torque_demand_a	INT16	RO	YES					●		0.1% Tn
6077	--	VAR	torque_actual_value_a	INT16	RO	YES		●	●		●		0.1% Tn
607A	--	VAR	target_position_a	INT32	RW	YES			●				position units
607C	--	VAR	home_offset_a	INT32	RW	YES	●		●			●	position units
607D	--	ARRAY	Software_position_limit_a	--	--	--			●				--
	0		number_of_entries	UINT8	RW	NO			●				--
	1		Min_position_limit_a	INT32	RW	NO			●				position units
	2		Max_position_limit_a	INT32	RW	NO			●				position units
607F	--	VAR	Max_profile_velocity_a	UINT32	RW	YES			●	●			speed units
6081	--	VAR	profile_velocity_a	UINT32	RW	YES			●				speed units
6082	--	VAR	end_velocity_a	UINT32	RW	YES			●				speed units
6083	--	VAR	profile_acceleration_a	UINT32	RW	YES			●	●			acceleration units
6084	--	VAR	profile_deceleration_a	UINT32	RW	YES			●	●			acceleration units
6085	--	VAR	quick_stop_deceleration_a	UINT32	RW	YES			●	●			acceleration units
6086	--	VAR	motion_profile_type_a	INT16	RW	YES			●				--
6087	--	VAR	torque_slope_a	UINT32	RW	YES					●		0.1%Tn/S
6093	--	ARRAY	position_factor_a	--	--	--	●		●		●	●	--
	0		number_of_entries_a	UINT32	RW	NO	●		●		●	●	--
	1		numerator_a	UINT32	RW	NO	●		●		●	●	--
	2		divisor_a	UINT32	RW	NO	●		●		●	●	--
6094	--	ARRAY	velocity_encoder_factor_a	--	--	--	●						--
	0		number_of_entries	UINT32	RW	NO	●						--
	1		numerator_a	UINT32	RW	NO	●						--
	2		divisor_a	UINT32	RW	NO	●						

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
6097	--	ARRAY	acceleration_factor_a	--	--	--	●						--
	0		number_of_entries_a	UINT32	RW	NO	●						--
	1		numerator_a	UINT32	RW	NO	●						--
	2		divisor_a	UINT32	RW	NO	●						--
6098	--	VAR	homing_method_a	INT8	RW	YES						●	

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
6099	--	ARRAY	homing_speeds_a	--	--	--						●	speed units
	0		number_of_entries	UINT8	RW	YES						●	
	1		speed_during_search_for_switch_a	UINT32	RW	YES						●	speed units
	2		speed_during_search_for_zero_a	UINT32	RW	YES						●	speed units
609A	--	VAR	homing_acceleration_a	UINT32	RW	YES						●	acceleration units
60A3	00	VAR	profile_jerk_use_a	UINT8	RO	NO			●				--
60A4	--	ARRAY	profile_jerk_a	--	--	--			●				--
	00		number of entries	UINT8	RO	NO			●				--
	01		profile_jerk1_a	UINT32	RW	NO			●				jerk unit
60C0	--	VAR	Interpolation sub mode select_a	INT16	RW	NO		●					
60C1	--	ARRAY	Interpolation data record_a	--	--	--		●					
	0		number_of_entries	UINT8	--	NO		●					
	1		the first parameter of ip function f_{ip}(x1, .. xN) _a	see 60C0h	RW	YES		●					position units
	2		the second parameter of ip function f_{ip}(x1, .. xN) _a		RW	YES		●					position units
60C2	--	RECORD	Interpolation time period_a	--	--	--		●					
	0		number_of_entries_a	UINT8	RO	NO		●					
	1		ip time units_a	UINT8	RW	NO		●					
	2		ip time index_a	UINT8	RW	NO		●					
60E0	--	VAR	Positive Torque Limit Value_a	UINT16	RW	YES	●						
60E1	--	VAR	Negtive Torque Limit Value_a	UINT16	RW	YES	●						
60F4	--	VAR	Following_error_actual_value_a	INT32	RO	YES						●	
60FA	--	VAR	control_effort_a	INT32	RO	YES		●	●	●		●	

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
60FC	--	VAR	position_demand_value*_a	INT32	RO	YES			●				inc
60FD	--	VAR	digital inputs_a	UINT32	RO	YES	●						
60FE	--	RECORD	digital outputs_a	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		pysical _outputs_a	UINT32	RWW	YES	●						
	2		bit_mask_a	UINT32	--	--	●						
60FF	--	VAR	target_velocity_a	UINT32	RW	YES				●			speed units
Those objects as below are applied to EDS/ETS only.													
683F	--	VAR	error_code_b	UINT16	RO	YES	●						--
6840	--	VAR	controlword_b	UINT16	RW	YES	●						--
6841	--	VAR	statusword_b	UINT16	RO	YES	●						--
685A	--	VAR	quick_stop_option_code_b	INT16	RW	NO	●						--
685B	--	VAR	shutdown_option_code_b	INT16	RW	NO	●						--
685C	--	VAR	disable_operation_option_code_b	INT16	RW	NO	●						--
685D	--	VAR	stop_option_code_b	INT16	RW	NO	●						--
685E	--	VAR	fault_reaction_option_code_b	INT16	RW	NO	●						--
6860	--	VAR	modes_of_operation_b	INT8	RW	YES	●						--
6861	--	VAR	modes_of_operation_display_b	INT8	RO	YES	●						--
6862	--	VAR	position_demand_value_b	INT32	RO	YES			●				position units
6863	--	VAR	position_actual_value*_b	INT32	RO	YES			●				inc
6864	--	VAR	position_actual_value_b	INT32	RO	YES		●	●		●	●	position units
6865	--	VAR	following_error_window_b	UINT32	RW	YES			●				position units
6866	--	VAR	following_error_time_out_b	UINT16	RW	YES			●				ms
6867	--	VAR	position_window_b	UINT32	RW	YES		●	●				position units
6868	--	VAR	position_window_time_b	UINT16	RW	YES		●	●				ms

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
6869	--	VAR	velocity_sensor_actual_value_b	UINT32	RO	YES				●			speed units
686B	--	VAR	velocity_demand_value_b	INT32	RO	YES				●			speed units
686C	--	VAR	velocity_actual_value_b	INT32	RO	YES	●						speed units
686D	--	VAR	velocity_window_b	UINT16	RW	YES				●			speed units
686E	--	VAR	velocity_window_time_b	UINT16	RW	YES				●			ms
686F	--	VAR	velocity_threshold_b	UINT16	RW	YES				●			speed units
6870	--	VAR	velocity_threshold_time_b	UINT16	RW	YES				●			ms
6871	--	VAR	target_torque_b	INT16	RW	YES					●		0.1% Tn
6872	--	VAR	Max_torque_b	UINT16	RW	YES					●		0.1% Tn
6874	--	VAR	torque_demand_b	INT16	RO	YES					●		0.1% Tn
6877	--	VAR	torque_actual_value_b	INT16	RO	YES		●	●		●		0.1% Tn
687A	--	VAR	target_position_b	INT32	RW	YES			●				position units
687C	--	VAR	home_offset_b	INT32	RW	YES		●	●			●	position units
687D	--	ARRAY	Software_position_limit_b	--	--	--			●				--
	0		number_of_entries	UINT8	RW	NO			●				--
	1		Min_position_limit_b	INT32	RW	NO			●				position units
	2		Max_position_limit_b	INT32	RW	NO			●				position units
687F	--	VAR	Max_profile_velocity_b	UINT32	RW	YES			●	●			speed units
6881	--	VAR	profile_velocity_b	UINT32	RW	YES			●				speed units
6882	--	VAR	end_velocity_b	UINT32	RW	YES			●	●			speed units
6883	--	VAR	profile_acceleration_b	UINT32	RW	YES			●	●			acceleration units
6084	--	VAR	profile_deceleration_b	UINT32	RW	YES			●	●			acceleration units
6085	--	VAR	quick_stop_deceleration_b	UINT32	RW	YES			●	●			acceleration units
6886	--	VAR	motion_profile_type_b	INT16	RW	YES			●				--
6887	--	VAR	torque_slope_b	UINT32	RW	YES					●		0.1%Tn/S

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
6893	--	ARRAY	position_factor_b	--	--	--		●	●	●		●	--
	0		number_of_entries	UINT32	RW	NO		●	●	●		●	--
	1		numerator_b	UINT32	RW	NO		●	●	●		●	--
	2		divisor_b	UINT32	RW	NO		●	●	●		●	--
6894	--	ARRAY	velocity_encoder_factor_b	--	--	--	●						--
	0		number_of_entries	UINT32	RW	NO	●						--
	1		numerator_b	UINT32	RW	NO	●						--
	2		divisor_b	UINT32	RW	NO	●						
6897	--	ARRAY	acceleration_factor_b	--	--	--	●						--
	0		number_of_entries	UINT32	RW	NO	●						--
	1		numerator_b	UINT32	RW	NO	●						--
	2		divisor_b	UINT32	RW	NO	●						--
6898	--	VAR	homing_method_b	INT8	RW	YES						●	
6899	--	ARRAY	homing_speeds_b	--	--	--						●	speed units
	0		number_of_entries	UINT8	RW	YES						●	
	1		speed_during_search_for_switch_b	UINT32	RW	YES						●	speed units
	2		speed_during_search_for_zero_b	UINT32	RW	YES						●	speed units
689A	--	VAR	homing_acceleration_b	UINT32	RW	YES						●	acceleration units
68A3	00	VAR	profile_jerk_use_b	UINT8	RO	NO			●				--
68A4	--	ARRAY	profile_jerk_b	--	--	--			●				--
	00		number of entries	UINT8	RO	NO			●				--
	01		profile_jerk1_b	UINT32	RW	NO			●				jerk unit
68C0	--	VAR	Interpolation sub mode select_b	INT16	RW	NO		●					
68C1	--	ARRAY	Interpolation data record_b	--	--	--		●					
	0		number_of_entries	UINT8	--	NO		●					

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
	1		the first parameter of ip function f_{ip}(x1, .. xN) _b	see 60C0h	RW	YES		●					position units
	2		the second parameter of ip function f_{ip}(x1, .. xN) _b		RW	YES		●					position units
68C2	--	RECORD	Interpolation time period_b	--	--	--		●					
	0		number_of_entries	UINT8	RO	NO		●					
	1		ip time units_b	UINT8	RW	NO		●					
	2		ip time index_b	UINT8	RW	NO		●					
68E0	--	VAR	Positive Torque Limit Value_b	UINT16	RW	YES	●						
68E1	--	VAR	Negtive Torque Limit Value_b	UINT16	RW	YES	●						
68F4	--	VAR	Following_error_actual_value_b	INT32	RO	YES						●	
68FA	--	VAR	control_effort_b	INT32	RO	YES		●	●	●		●	
68FC	--	VAR	position_demand_value*_b	INT32	RO	YES			●				inc
68FD	--	VAR	digital inputs_b	UINT32	RO	YES	●						
68FE	--	RECORD	digital outputs_b	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		pysical _outputs_b	UINT32	RWW	YES	●						
	2		bit_mask_b	UINT32	--	--	●						
68FF	--	VAR	target_velocity_b	UINT32	RW	YES				●			speed units
Those objects as below are applied to ETS only.													
703F	--	VAR	error_code_c	UINT16	RO	YES	●						--
7040	--	VAR	controlword_c	UINT16	RW	YES	●						--
7041	--	VAR	statusword_c	UINT16	RO	YES	●						--
705A	--	VAR	quick_stop_option_code_c	INT16	RW	NO	●						--
705B	--	VAR	shutdown_option_code_c	INT16	RW	NO	●						--

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
705C	--	VAR	disable_operation_option_code_c	INT16	RW	NO	●						--
705D	--	VAR	stop_option_code_c	INT16	RW	NO	●						--
705E	--	VAR	fault_reaction_option_code_c	INT16	RW	NO	●						--
7060	--	VAR	modes_of_operation_c	INT8	RW	YES	●						--
7061	--	VAR	modes_of_operation_display_c	INT8	RO	YES	●						--
7062	--	VAR	position_demand_value_c	INT32	RO	YES			●				position units
7063	--	VAR	position_actual_value*_c	INT32	RO	YES			●				inc
7064	--	VAR	position_actual_value_c	INT32	RO	YES		●	●		●	●	position units
7065	--	VAR	following_error_window_c	UINT32	RW	YES			●				position units
7066	--	VAR	following_error_time_out_c	UINT16	RW	YES			●				ms
7067	--	VAR	position_window_c	UINT32	RW	YES		●	●				position units
7068	--	VAR	position_window_time_c	UINT16	RW	YES		●	●				ms
7069	--	VAR	velocity_sensor_actual_value_c	UINT32	RO	YES				●			speed units
706B	--	VAR	velocity_demand_value_c	INT32	RO	YES				●			speed units
706C	--	VAR	velocity_actual_value_c	INT32	RO	YES	●						speed units
706D	--	VAR	velocity_window_c	UINT16	RW	YES				●			speed units
706E	--	VAR	velocity_window_time_c	UINT16	RW	YES				●			ms
706F	--	VAR	velocity_threshold_c	UINT16	RW	YES				●			speed units
7070	--	VAR	velocity_threshold_time_c	UINT16	RW	YES				●			ms
7071	--	VAR	target_torque_c	INT16	RW	YES					●		0.1% Tn
7072	--	VAR	Max_torque_c	UINT16	RW	YES					●		0.1% Tn
7074	--	VAR	torque_demand_c	INT16	RO	YES					●		0.1% Tn
7077	--	VAR	torque_actual_value_c	INT16	RO	YES		●	●		●		0.1% Tn
707A	--	VAR	target_position_c	INT32	RW	YES			●				position units
707C	--	VAR	home_offset_c	INT32	RW	YES		●	●			●	position units

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
707D	--	ARRAY	Software_position_limit_c	--	--	--			●				--
	0		number_of_entries	UINT8	RW	NO			●				--
	1		Min_position_limit_c	INT32	RW	NO			●				position units
	2		Max_position_limit_c	INT32	RW	NO			●				position units
707F	--	VAR	Max_profile_velocity_c	UINT32	RW	YES			●	●			speed units
7081	--	VAR	profile_velocity_c	UINT32	RW	YES			●				speed units
7082	--	VAR	end_velocity_c	UINT32	RW	YES			●	●			speed units
7083	--	VAR	profile_acceleration_c	UINT32	RW	YES			●	●			acceleration units
7084	--	VAR	profile_deceleration_c	UINT32	RW	YES			●	●			acceleration units
7085	--	VAR	quick_stop_deceleration_c	UINT32	RW	YES			●	●			acceleration units
7086	--	VAR	motion_profile_type_c	INT16	RW	YES			●				--
7087	--	VAR	torque_slope_c	UINT32	RW	YES					●		0.1%Tn/S
7093	--	ARRAY	position_factor_c	--	--	--		●	●	●		●	--
	0		number_of_entries	UINT32	RW	NO		●	●	●		●	--
	1		numerator_c	UINT32	RW	NO		●	●	●		●	--
	2		divisor_c	UINT32	RW	NO		●	●	●		●	--
7094	--	ARRAY	velocity_encoder_factor_c	--	--	--	●						--
	0		number_of_entries	UINT32	RW	NO	●						--
	1		numerator_c	UINT32	RW	NO	●						--
	2		divisor_c	UINT32	RW	NO	●						
7097	--	ARRAY	acceleration_factor_c	--	--	--	●						--
	0		number_of_entries	UINT32	RW	NO	●						--
	1		numerator_c	UINT32	RW	NO	●						--
	2		divisor_c	UINT32	RW	NO	●						--
7098	--	VAR	homing_method_c	INT8	RW	YES						●	

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
7099	--	ARRAY	homing_speeds_c	--	--	--						●	speed units
	0		number_of_entries	UINT8	RW	YES						●	
	1		speed_during_search_for_switch_c	UINT32	RW	YES						●	speed units
	2		speed_during_search_for_zero_c	UINT32	RW	YES						●	speed units
709A	--	VAR	homing_acceleration_c	UINT32	RW	YES						●	acceleration units
70A3	00	VAR	profile_jerk_use_c	UINT8	RO	NO			●				--
70A4	--	ARRAY	profile_jerk_c	--	--	--			●				--
	00		number of entries	UINT8	RO	NO			●				--
	01		profile_jerk1_c	UINT32	RW	NO			●				jerk unit
70C0	--	VAR	Interpolation sub mode select_c	INT16	RW	NO		●					
70C1	--	ARRAY	Interpolation data record_c	--	--	--		●					
	0		number_of_entries	UINT8	--	NO		●					
	1		the first parameter of ip function f_{ip}(x1, .. xN) _c	see 60C0h	RW	YES		●					position units
	2		the second parameter of ip function f_{ip}(x1, .. xN) _c		RW	YES		●					position units
70C2	--	RECORD	Interpolation time period_c	--	--	--		●					
	0		number_of_entries	UINT8	RO	NO		●					
	1		ip time units_c	UINT8	RW	NO		●					
	2		ip time index_c	UINT8	RW	NO		●					
70E0	--	VAR	Positive Torque Limit Value_c	UINT16	RW	YES	●						
70E1	--	VAR	Negtive Torque Limit Value_c	UINT16	RW	YES	●						
70F4	--	VAR	Following_error_actual_value_c	INT32	RO	YES						●	
70FA	--	VAR	control_effort_c	INT32	RO	YES		●	●	●		●	

Index	Subindex	Object	Name	Type	Attr.	PDO	Support						Unit
							All	IP	PP	PV	PT	HM	
70FC	--	VAR	position_demand_value*_c	INT32	RO	YES			●				inc
70FD	--	VAR	digital inputs_c	UINT32	RO	YES	●						
70FE	--	RECORD	digital outputs_c	--	--	--	●						
	0		number_of_entries	UINT8	RO	NO	●						
	1		pysical _outputs_c	UINT32	RWW	YES	●						
	2		bit_mask_c	UINT32	--	--	●						
70FF	--	VAR	target_velocity_c	UINT32	RW	YES				●			speed units

Note: * is used for ProNet-□□AMG only.

ESTUN AUTOMATION TECHNOLOGY CO., LTD

Address: 16 Shuige Road, Jiangning Development Zone
Nanjing 211106, P.R.China
Tel: +86-25-58328505/8507
Fax: +86-25-58328504
Web: www.estun.cn
E-mail: export@estun.com

www.estun.cn

ESTUN