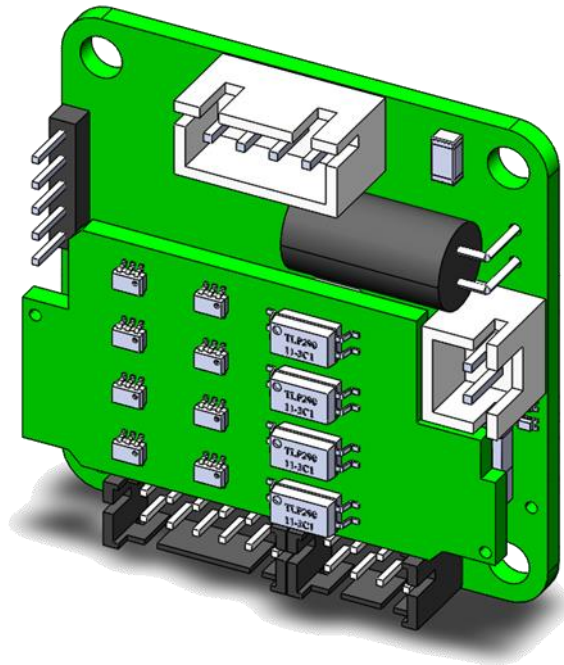


# CANopen firmware manual EZD-1136

Order command : EZD-1136



## Revisions

Version	Modifications	Writer	Checker	Date
1.0	Initial version	MLE	SRU	04/11/2024
1.1	Add dictionary object	MLE	SRU	11/12/2024
1.2	Add emergency error	MLE	SRU	19/12/2024
1.3	Add information in dictionary object	MLE	SRU	03/01/2025
1.4	Add prerequisites information	MLE	SRU	13/01/2025

## Table of contents

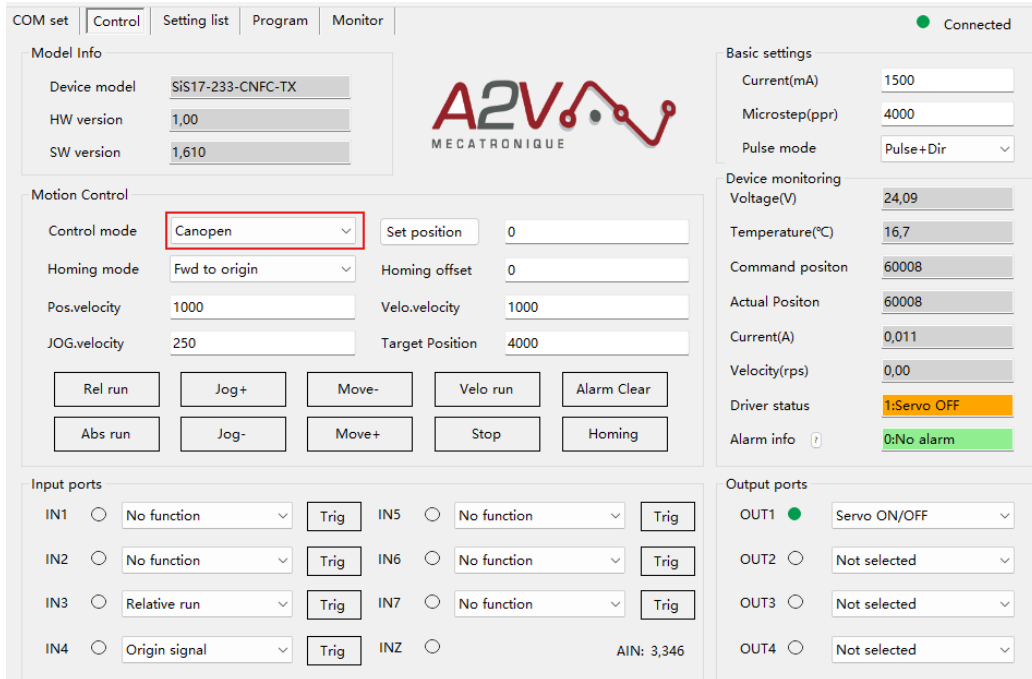
<b>1.</b>	<b><i>CAN-open Communication object identifier</i></b> .....	<b>3</b>
•	<b>CAN-open communicate service</b> .....	<b>4</b>
•	<b>CAN-open Communication object identifier</b> .....	<b>4</b>
•	<b>Object Dictionary (OD)</b> .....	<b>5</b>
▪	Overview of Object Dictionary .....	5
▪	Accessing Properties.....	6
▪	Communication Object Dictionary .....	6
•	<b>Network Management (NMT)</b> .....	<b>6</b>
▪	NMT error control .....	7
▪	Node/Life Protection .....	7
▪	Herthbeat .....	8
▪	Communication state machine.....	10
•	<b>Service Data Object (SDO)</b> .....	<b>10</b>
•	<b>Process Data Objects (PDO)</b> .....	<b>11</b>
▪	PDO The transmission framework and characteristics of.....	11
▪	PDO object.....	14
▪	PDO communication parameters .....	14
▪	PDO Mapping parameters .....	16
•	<b>Synchronization objects (SYNC)</b> .....	<b>16</b>
▪	Synchronous generator .....	17
▪	Transmission framework for synchronous objects.....	17
•	<b>Emergency Object Services (EMCY)</b> .....	<b>18</b>
▪	Emergency error code .....	19
▪	Operation status.....	20
<b>2.</b>	<b><i>CAN parameter description and settings</i></b> .....	<b>20</b>
•	<b>SDO parameter list</b> .....	<b>20</b>
▪	Configuration parameters .....	20
▪	Motion parameters .....	22
•	<b>Essential functions</b> .....	<b>24</b>
▪	Operatings modes .....	24
▪	Control and status word .....	26
•	<b>Default mapping PDO</b> .....	<b>28</b>

## 1. Prerequisites

- **Config simple tuner**

To start motion in CANopen be sure to have the good configuration in simpleTuner.

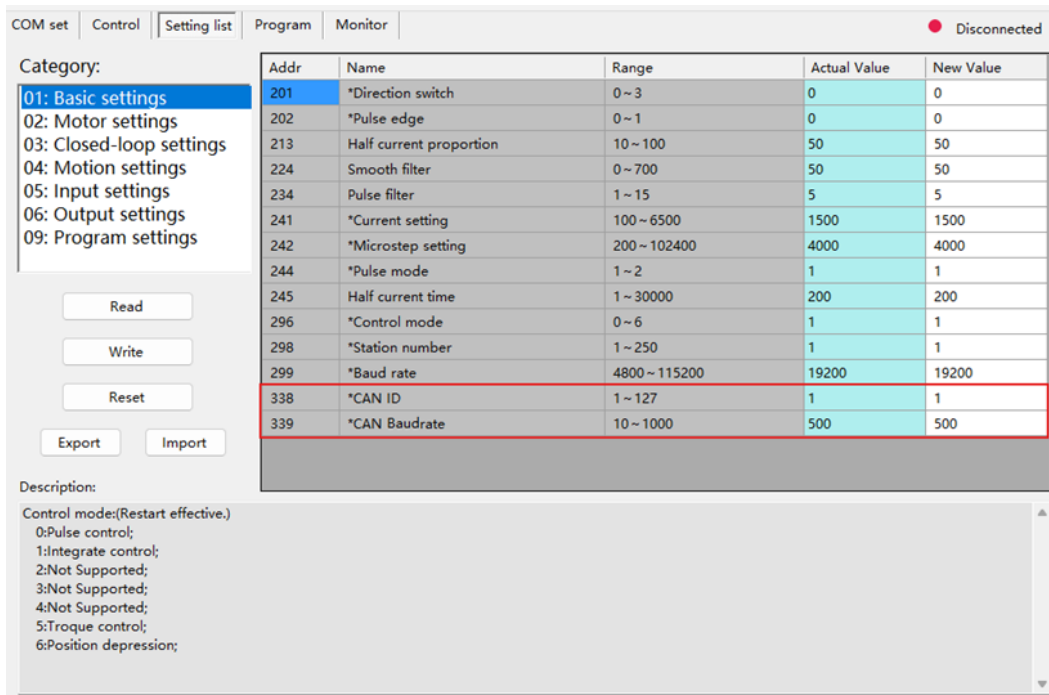
First the control mode must be set to Canopen :



The screenshot shows the 'Control' tab in the simpleTuner software. The 'Control mode' dropdown menu is set to 'Canopen' and is highlighted with a red box. Other settings visible include:

- Model Info: Device model SIS17-233-CNFC-TX, HW version 1,00, SW version 1,610.
- Motion Control: Control mode Canopen, Homing mode Fwd to origin, Pos.velocity 1000, JOG.velocity 250, Set position 0, Homing offset 0, Velo.velocity 1000, Target Position 4000.
- Basic settings: Current(mA) 1500, Microstep(ppr) 4000, Pulse mode Pulse+Dir.
- Device monitoring: Voltage(V) 24,09, Temperature(°C) 16,7, Command position 60008, Actual Position 60008, Current(A) 0,011, Velocity(rps) 0,00.
- Driver status: 1:Servo OFF.
- Alarm info: 0:No alarm.

Second, be sure that the CAN ID, CAN baud rate match with the CAN master :



The screenshot shows the 'Setting list' tab in the simpleTuner software. The table below lists various parameters, with rows 338 and 339 highlighted in red.

Addr	Name	Range	Actual Value	New Value
201	*Direction switch	0 ~ 3	0	0
202	*Pulse edge	0 ~ 1	0	0
213	Half current proportion	10 ~ 100	50	50
224	Smooth filter	0 ~ 700	50	50
234	Pulse filter	1 ~ 15	5	5
241	*Current setting	100 ~ 6500	1500	1500
242	*Microstep setting	200 ~ 102400	4000	4000
244	*Pulse mode	1 ~ 2	1	1
245	Half current time	1 ~ 30000	200	200
296	*Control mode	0 ~ 6	1	1
298	*Station number	1 ~ 250	1	1
299	*Baud rate	4800 ~ 115200	19200	19200
338	*CAN ID	1 ~ 127	1	1
339	*CAN Baudrate	10 ~ 1000	500	500

Description:  
Control mode:(Restart effective.)  
0:Pulse control;  
1:Integrate control;  
2:Not Supported;  
3:Not Supported;  
4:Not Supported;  
5:Troque control;  
6:Position depression;

For more information about simple tuner software, please use its documentation.

## 2. CAN-open Communication object identifier

- **CAN-open communicate service**

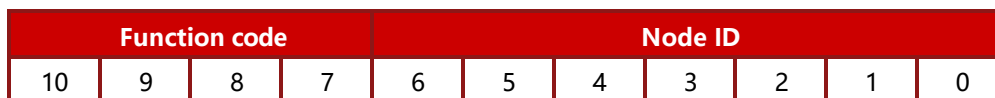
The product follows CAN-open Specification :

- Follow CAN 2.0A standard
- Compliant with CAN-open standard protocol DS 301 V4.02
- Compliant with CAN-open standard protocol DSP 402 V2.01
- Services supported by CANOpen drives:
- Support NMT Slave services
- Equipment monitoring: supports heartbeat messages and node protection
- Support for PDO services: Each slave station can be configured with up to 4 TxPDO and 4 RxPDO
- PDO transmission type: supports event triggering, time triggering, synchronous cycle, synchronous non cycle
- Support SDO services
- Support Emergency Protocol

- **CAN-open Communication object identifier**

The Communication Object identifier (COB-ID) is used to specify the priority of the communication object and its recognition.

11 bit CAN ID, including a 4-bit function code section and a 7-bit Node ID section, as shown in the following figure :



The range of Node ID is 1-127 (0 is not allowed to be used).

Each communication object of CANopen has a default COB-ID that can be read through SDO, and some can be modified through SDO. right

The list of objects is as follows :

CAN-open Predefine broadcast objects for master/slave connection sets				
Object	Function code	Node AD	COB-ID	Object Dictionary Index
NMT Network Management	0000	0	0x000	-
Synchronization Object	0001	0	0x080	1005H,1006H,1007H
emergent	0001	1~127	0x081~0x0FF	1024H,1015H
TXPDO1(sending)	0011	1~127	0x181~0x1FF	1800H
RXPDO1(receive)	0100	1~127	0x201~0x27F	1400H
TXPDO2(sending)	0101	1~127	0x281~0x2FF	1801H
RXPDO2 (receive)	0110	1~127	0x301~0x37F	1401H

TXPDO3(sending)	0111	1~127	0x381~0x3FF	1802H
-----------------	------	-------	-------------	-------

**Attention :**

- 1) PDO/SDO sending/receiving is relative to the slave CAN node.
- 2) NMT error control includes Node Guarding, Heartbeat, and Boot-up protocols.
- 3) The ID address allocation table corresponds to a predefined set of master-slave connections, as all peer IDs are different, there is actually only one master device

All connected node IDs can communicate with each connected slave node (up to 127) in a peer-to-peer manner. Two connected slave nodes cannot communicate. PDO/SDO

- **Object Dictionary (OD)**

- **Overview of Object Dictionary**

An object dictionary is an ordered set of objects; Each object is addressed using a 16 bit index value to allow access

A single element in a data structure is defined with an 8-bit subindex, and the structure of the object dictionary is shown in the table below :

INDEX	OBJECT
0000H	not used
0001H—001FH	Standard data types, such as Boolean, signed sixteen bit (Integer16), etc
0020H—003FH	Complex data types, such as PDO communication parameters (PDOCompar), etc
0040H—005FH	The responsible data type specified by the manufacturer
0060H—007FH	Standard data types specified in the device sub protocol
0080H—009FH	Complex data types specified by device sub protocols
00A0H—0FFFH	reserved area
1000H—1FFFH	Communication sub protocol area, such as device type, number of PDOs, etc
2000H—5FFFH	Manufacturer specific sub protocol area
6000H—9FFFH	Standard device sub protocol areas, such as the object dictionary area of DSP 402

The detailed definition of an object dictionary is described in an electronic data document (EDS), which can be obtained by contacting our technical team.

The description of the three main object dictionaries in EDS is as follows:

- (1) Communication object dictionaries, such as object dictionaries for 1000H, 1400H, 1A00H, etc...
- (2) Manufacturer custom object dictionary, such as 2000H-2130H.
- (3) CIA DSP402 Partial Object Dictionary.

- Accessing Properties

attribute	description
RW	Read/Write
WO	Write only
RO	Read only
CO	Constant, Read
RWW	Read/Write on process output

- Communication Object Dictionary

The list of communication object dictionaries is as follows :

Index	Name	Data type	Acces
0x1000	Device Type	U32	RO
0x1001	Error Register	U8	RO
0x1005	SYNC COB ID	U32	RW
0x1006	Communication Cycle Period	U32	RW
0x1008	Manufacturer Device Name	Const	CO
0x1009	Manufacturer Hardware Version	Const	CO
0x100A	Manufacturer Software Version	Const	CO
0x100C	Guard Time	U16	RW
0x1014	Emergency COB ID	U32	RW
0x1017	Producer Heartbeat Time	U16	RW
0x1200	Server SDO Parameter	Server SDO parameters	RO
0x1400	Receive PDO 1 Parameter	Receive PDO parameters	RW
0x1401	Receive PDO 2 Parameter	Receive PDO parameters	RW
0x1402	Receive PDO 3 Parameter	Receive PDO parameters	RW
0x1403	Receive PDO 4 Parameter	Receive PDO parameters	RW
0x1600	Receive PDO 1 Mapping	Receive PDO mapping	RW
0x1601	Receive PDO 2 Mapping	Receive PDO mapping	RW
0x1602	Receive PDO 3 Mapping	Receive PDO mapping	RW
0x1603	Receive PDO 4 Mapping	Receive PDO mapping	RW
0x1800	Transmit PDO 1 Parameter	Transmit PDO parameters	RW
0x1801	Transmit PDO 2 Parameter	Transmit PDO parameters	RW
0x1802	Transmit PDO 3 Parameter	Transmit PDO parameters	RW
0x1803	Transmit PDO 4 Parameter	Transmit PDO parameters	RW
0x1A00	Transmit PDO 1 Mapping	Transmit PDO mapping	RW
0x1A01	Transmit PDO 2 Mapping	Transmit PDO mapping	RW
0x1A02	Transmit PDO 3 Mapping	Transmit PDO mapping	RW
0x1A03	Transmit PDO 4 Mapping	Transmit PDO mapping	RW

- **Network Management (NMT)**

NMT provides network management services. This service is implemented using a master-slave communication mode (so there is only one NMT master node).

Only the NMT master node can transmit NMT module control messages, and all slave nodes must support NMT module control services. NMT module control does not require a response. The message format is as follows: NMT master node → NMT slave node

COB-ID	Byte 0	Byte 1
0x000	command	Node-ID

When Node ID=0, all NMT slave nodes are addressed. The correspondence between the values of command words and services is shown in the table below:

Command	NMT serveurur
1(01H)	Start remote node
2(02H)	Stop remote node
128(80H)	Enter pre operation state
129(81H)	Node reset
130(82H)	Communication reset

- NMT error control

NMT error control is mainly used to detect whether devices in the network are online and the status of the devices, including node protection, lifespan protection, and heartbeat.

**Warning:**

- 1) Node protection and heartbeat cannot be used simultaneously;
- 2) Node protection, the heartbeat time cannot be set too short to avoid increasing network load.

- Node/Life Protection

Node protection refers to the NMT host periodically querying the NMT slave status through remote frames; Lifetime protection refers to the monitoring of the slave station received by the slave station

Remote frame intervals are used to indirectly monitor the status of the master station, and node protection follows the master-slave model, where each remote frame must receive a response.

The objects related to node/lifespan protection include a protection time of 100Ch and a lifespan factor of 100Dh. The value of 100Ch is for node protection to be far away under normal circumstances

The interval between frames, measured in milliseconds, is determined by the product of 100Ch and 100Dh, which determines the latest query time for the host. Under normal circumstances, node protection can be achieved. When nodes 100Ch and 100Dh are non-0 and receive a node protection request frame, lifespan protection is activated.

The master station sends node protection remote frames every 100Ch, and the slave must respond, otherwise it is considered disconnected from the slave station; Time from station 100Ch \* 100Dh

If the node protection remote frame is not received, it is considered that the main station is disconnected. Through this service, the NMT master node can check the current status of each node, and the master node sends remote frames in the following format:

NMT master node → NMT slave node

COB-ID
0x700+Node-ID

The format of the NMT response message from the node is as follows:

NMT slave node → NMT master node

COB-ID	Byte 0
0x700+Node-ID	Bit 6:0 state

The data section includes a trigger bit (bit7), which must be alternately set to "0" or "1" in each node protection response. The trigger bit is set to "0" on the first node protection request. Bits 0 to 6 represent the node state, and the corresponding relationship between their values and the state is shown in the table below:

Value	State
0(00H)	initialization
1(01H)	not connected
2(02H)	connected
3(03H)	prepare
4(04H)	cease
5(05H)	operate
127(7FH)	Pre operation

## ▪ Herthbeat

A node can be configured to generate periodic packets called heartbeat. The heartbeat mode adopts a producer consumer model. CANopen devices can send heartbeat messages based on the cycle set by the producer heartbeat interval object 1017h, in milliseconds. A node in the network that always has consumer heartbeat function monitors the producer based on the consumer time set by object 1016h. If the producer's heartbeat of the corresponding node is not received within the consumer heartbeat time range, it is considered disconnected (or faulty).

After configuring the producer heartbeat interval of 1017h, the node heartbeat function is activated and starts to generate heartbeat messages. After configuring a valid sub index for consumer heartbeat of 1016 hours, monitoring begins upon receiving a frame of heartbeat from the corresponding node.

If the host sends a heartbeat message according to its producer time and the slave of the monitoring host does not receive the heartbeat message within 1016h sub index time, it is considered that the host has dropped. 1016h, a sub index time  $\geq$  host producer time  $\times 2$ . Otherwise, it is easy to mistakenly report that the slave machine is believed to have dropped the host.



The slave sends a heartbeat message every 1017 hours to monitor the master (or other slave) of the slave. If the heartbeat message is not received within the consumer's time, it is considered that the slave has dropped. 1017h

$\times 2 \leq$  Monitor the consumer time of the host (or other slave) of the slave, otherwise it is easy to misreport the slave dropping.

The format of the heartbeat message is shown in the table. The data segment only contains one byte, while the rest are consistent with the status of the node protection response message in the table.

Heartbeat Producer → Consumer

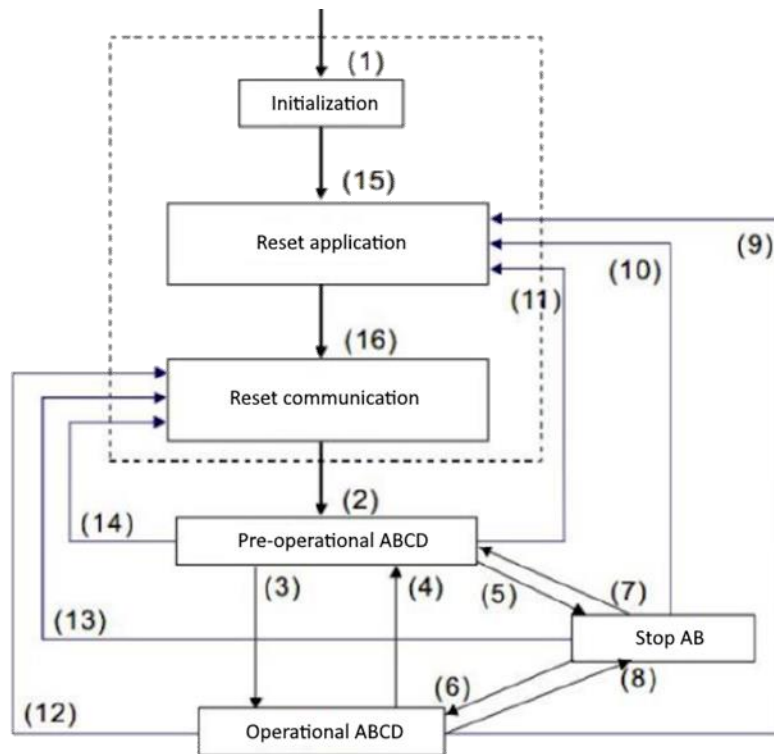
COB-ID	Byte 0
0x700+Node-ID	state

The corresponding meanings of its values are shown in the table below:

State	Meaning
0(00H)	Initialization
4(04H)	cease
5(05H)	operate
127(7FH)	Pre operation

Note: The driver is a heartbeat producer. It is recommended that the heartbeat producer's time should not be less than 20ms.

▪ Communication state machine



**Explanation:**

(1) After the power is turned on, it automatically enters the initialization state A: NMT

(2) Automatically enter pre operation state B: Node Guard

(3)(6) Start remote node C: SDO

(4)(7) Enter pre operation state D: Emergency

(5)(8) Stop remote node E: PDO

(9) (10) (11) Reset node F: Boot up

(12) (13) (14) Reset communication

(15) Automatically enter the reset application state

(16) Automatically enter reset communication status

After device initialization (collectively referred to as initialization, reset application, and reset communication in the figure) is completed, it enters the pre operation state. Devices in this state can be accessed through

SDO (such as using configuration tools) sets parameters and assigns IDs. Then, the node directly enters the operational state

• **Service Data Object (SDO)**

SDO is used to access the object dictionary of a device. The visitor is referred to as the client, and the object dictionary is accessed to provide the requested service

CAN-open devices are also known as servers. The customer's CAN message and the server's response CAN message always contain 8 bytes of data (although not all data bytes are necessarily meaningful). A customer's request must receive a response from the server.

Its basic structure is as follows:

Customer → Server/Server → Customer

Byte 0	Byte 1:2	Byte 3	Byte 4:7
SDO Command	Object Index	Object subindex	data

Example: Using an SDO message to write the value 0x20F0 to the object dictionary with ID 2, index 1801H, and sub index 3.

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Customer → Server								
602	2B	01	18	03	F0	20	00	00
Server → Customer								
582	60	01	18	03	00	00	00	00

Using the SDO message below, read out the data of the object with index 1801H and sub index 3 in the object dictionary.

COB-ID	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Customer → Server								
602	40	01	18	03	00	00	00	00
Server → Customer								
582	4B	01	18	03	F0	20	00	00

SDO clients or servers terminate SDO transmission by sending a message in the following format:

Customer → Server/Server → Customer

Bit	7	6	5	4	3	2	1	0
	1	0	0	-	-	-	-	-

In the SDO transmission termination message, data bytes 0 and 1 represent the object index, byte 2 represents the subindex, and bytes 4 to 7 contain 32-bit termination codes, which describe the reason for the message termination.

- **Process Data Objects (PDO)**

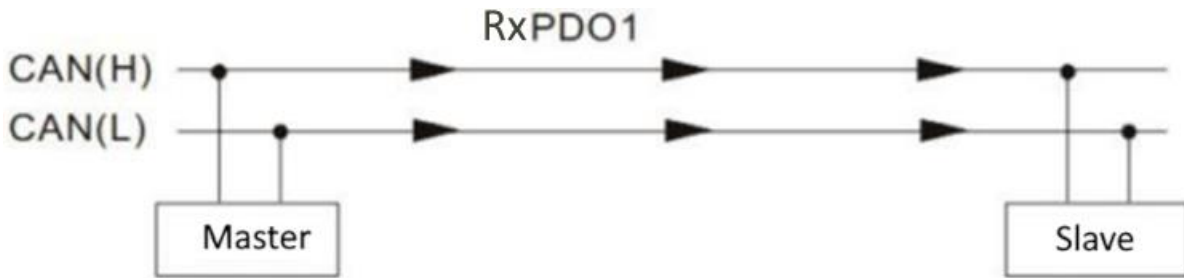
- **PDO The transmission framework and characteristics of**

Process Data Object (PDO) is used for real-time data transmission and is the primary data transmission method in CANopen. PDO adopts a producer/consumer mode, with a length of less than 8 bytes and a relatively fast transmission speed. PDO data transmission can be done in a one-to-one or one to many manner. Each PDO information includes both sending PDO (TxPDO) and receiving PDO (RxPDO) information, and its transmission method is defined in the PDO communication parameter index.

All PDO transmitted data must be mapped to the corresponding index area through an object dictionary. Taking the 1600H and 1A00H objects defined in DSP 402 as examples:

Note: The values in the object dictionary in the figure are for example only and do not have practical significance.

Master station sends information to slave station



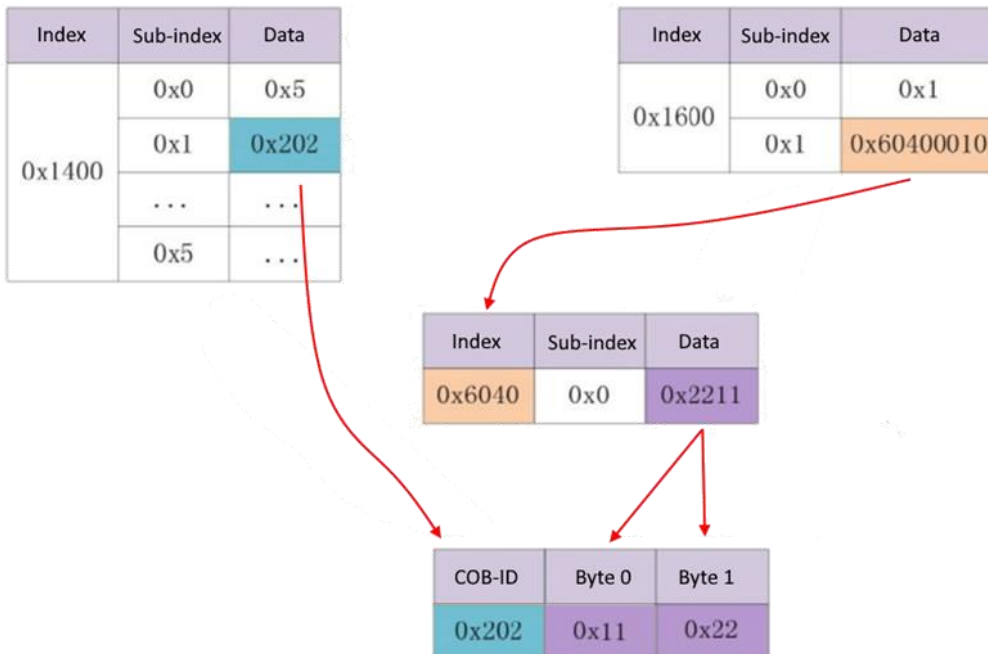
PDO1 data value Data 0, Data 1, Data 2, Data 3, Data 4, Data 5, Data 6, Data 7, 0x77, 0x88,

Index	Sub	Definition	Value	R/W	Size
0x1600	0	0. Number	1	R/W	U8
0x1600	1	1. Mapped Object	0x60400010	R/W	U32
0x1600	2	2. Mapped Object	0	R/W	U32
0x1600	3	3. Mapped Object	0	R/W	U32
0x1600	4	4. Mapped Object	0	R/W	U32
0x6040	0	0. Control word	0x2211	R/W	U16 (2 Byte)

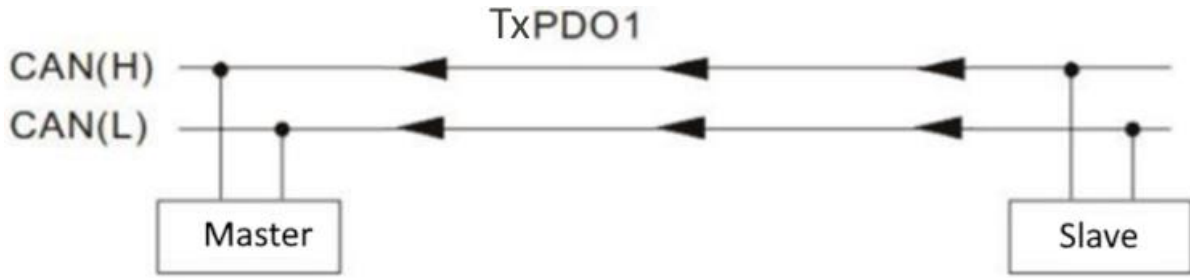
PDO1 Map

0x60400010

The following diagram provides a detailed description of the relationship between PDO parameters (1400H) and PDO mapping (1600H), as well as the transmission process of PDO data (using node 2 as an example). The direction of the arrow in the diagram represents the direction of data processing at the main station.



The main station receives information and the information returned from the station

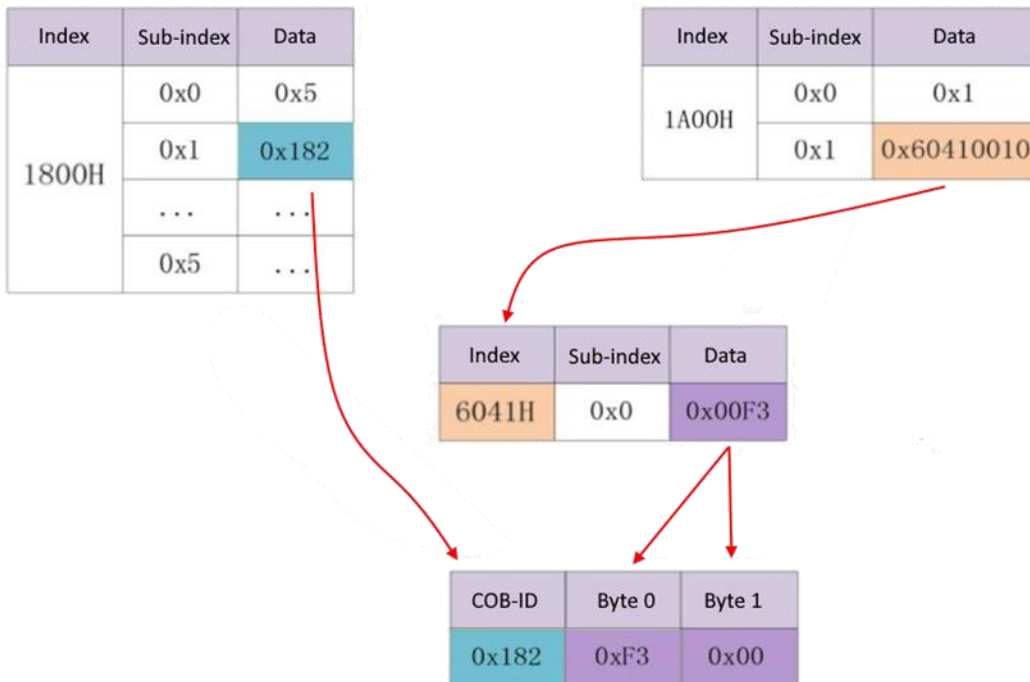


PDO1 data value Data 0, Data 1, Data 2, Data 3, Data 4, Data 5, Data 6, Data 7, 0xF3, 0x00,

Index	Sub	Definition	Value	R/W	Size
0x1A00	0	0. Number	1	R/W	U8
0x1A00	1	1. Mapped Object	0x60410010	R/W	U32
0x1A00	2	2. Mapped Object	0	R/W	U32
0x1A00	3	3. Mapped Object	0	R/W	U32
0x1A00	4	4. Mapped Object	0	R/W	U32
0x6041	0	Statusword	0xF3	R/W	U16

PDO1 Map

The following diagram provides a detailed description of the relationship between PDO parameters (1800H) and PDO mapping (1A00H), as well as the transmission process of PDO data (using node 2 as an example). The direction of the arrow in the diagram represents the direction of data processing from the station.



In this product, CANopen communication only supports point-to-point PDO transmission.

- PDO object

According to the different receiving and sending methods, please note that the description here is for the slave station. PDO is divided into TPDO and RPDO, using 4 TPDOs and 4 RPDOs. The list of related objects is as follows:

object	function code	COB-ID	Communication Object	Mapping Objects
TXPDO1(sending)	0011	0x181~0x1FF	1800H	1A00H
RXPDO1(receive)	0100	0x201~0x27F	1400H	1600H
TXPDO2(sending)	0101	0x281~0x2FF	1801H	1A01H
RXPDO2(receive)	0110	0x301~0x37F	1401H	1601H
TXPDO3(sending)	0111	0x381~0x3FF	1802H	1A02H
RXPDO3(receive)	1000	0x401~0x47F	1402H	1602H
TXPDO4(sending)	1001	0x481~0x4FF	1803H	1A03H
RXPDO4(receive)	1010	0x501~0x57F	1403H	1603H

- PDO communication parameters

- 1) CAN identifier for PDO

The CAN identifier of PDO, also known as the COB-ID of PDO, contains control bits and identification data to determine the bus priority of the PDO. The COB-ID is located on sub index 01 of the communication parameters (RPDO: 1400h~1403h, TPDO: 1800h~1803h), and the highest bit determines whether the PDO is valid.

The drive only supports point-to-point PDO transmission, so the lower 7 bits of COB-ID must be the station number address of the node.

- 2) PDO transmission type

The transmission type of PDO is located on sub index 02 of communication parameters (RPDO: 1400h~1403h, TPDO: 1800h~1803h)

Asynchronous transmission - triggered by events, including data change triggering and periodic event timer triggering;

Synchronous transmission - related to synchronous frames in a network.

Communication parameters (RPDO: 1400h~1403h, TPDO: 1800h~1803h) sub index 02 Different values represent different transmission types, defined

The method of triggering TPDO transmission or processing received RPDO is shown in the table for specific correspondence.

Communication type value	synchronous		asynchronous
	cycle	Non-cycle	

0		√	
1~240	√		
241~254	reserve		
254、 255			√

**Explanation:**

- 1) When the TPDO transmission type is 0, if the data of the mapped object changes and a synchronization frame is received, the TPDO is sent;
- 2) When the transmission type of TPDO is 1-240, when the corresponding number of synchronization frames are received, the TPDO is sent.
- 3) When the transmission type of TPDO is 254 or 255, if the mapping data changes or the event timer arrives, the TPDO is sent.
- 4) When the transmission type of RPDO is 0-240, as long as a synchronization frame is received, the latest data of that RPDO will be updated to the application; When the transmission type of RPDO is 254 or 255, the received data will be directly updated to the application.

2) Prohibited time

A prohibition time has been set for TPDO, stored on sub index 03 of communication parameters (1800h~1803h), to prevent the CAN network from being continuously occupied by lower priority PDO. The unit of this parameter is 100us. After setting the value, the transmission interval of the same TPDO should not be reduced by less than the time corresponding to this parameter.

For example, if the prohibition time of TPDO2 is 300, the transmission interval of TPDO will not be less than 30ms.

Suggestion: When frequently changing objects (such as feedback position, feedback speed, etc.) are configured to TPDO and the transmission type of the TPDO is asynchronous, it is recommended to set a certain prohibition time.

3) Event Timer

For TPDO with asynchronous transmission (transmission type 254 or 255), define an event timer located at sub index 05 of communication parameters (1800h~1803h). The event timer can also be seen as a triggering event, which will also trigger the corresponding TPDO transmission. If there are other events such as data changes during the running cycle of the timer, TPDO will also be triggered, and the event counter will be immediately reset.

4) Suggestion for configuring PDO attributes

(1) Synchronous or asynchronous: The synchronous transmission method refers to the update of the data corresponding to PDO when a synchronous frame is generated on the bus. Its characteristic is that the data update cycle is stable, but it cannot be synchronized with data changes in real time. Asynchronous refers to the process of updating data as soon as it changes. This transmission method responds quickly, but for frequently changing data (such as real-time location information), it can easily cause significant data load on the bus. Therefore, a prohibition time parameter is often configured (after unsuccessful data transmission, it should be sent at intervals instead of repeatedly and continuously) to reduce network load.

So it is recommended to use synchronous PDO for parameters with low real-time requirements within the network, and asynchronous PDO for parameters with high real-time requirements. However, attention should be paid to configuring prohibition time to protect the network load from impact.

(2) Setting of synchronization period: It is recommended to calculate according to the empirical formula (baud rate of 1M):

$$\text{Synchronous cycle (milliseconds)} = [\text{PDO total} / 9] / (40\%) + 2$$

Assuming a CANopen network has a total of 12 axes, each with a sending and receiving PDO. So the total number of PDOs is  $12 * 2 = 24$ . Within each millisecond, when the bus is at full load, approximately 9 PDOs can be transmitted. Considering the bus load margin, assuming a bus load of 40% (a relatively reasonable load rate), the required time for 24 PDO transmissions is:  $24 / 9 / (40\%) = 6.67$  (milliseconds). Considering the time cost of SDO, synchronization frames, heartbeat messages, emergency messages, etc. in the network, an additional 2 milliseconds are required. It is recommended to configure a synchronization cycle of 8.67 milliseconds.

The above empirical formula is also applicable to setting the prohibition time of asynchronous PDO.

- **PDO Mapping parameters**

The PDO mapping parameters contain pointers to the process data corresponding to the PDO that PDO needs to send or receive, including indexes, sub indexes, and the length of the mapping object. Each PDO data can have a maximum length of 8 bytes and can map one or more objects simultaneously. The sub index 0 records the specific number of objects mapped by the PDO, while sub indexes 1-8 represent the mapped content. The mapping parameter content is defined as follows.

<b>bit</b>	31	... ..	16	15	... ..	8	7	... ..	0
<b>define</b>	index			Sub-Index			Object length		

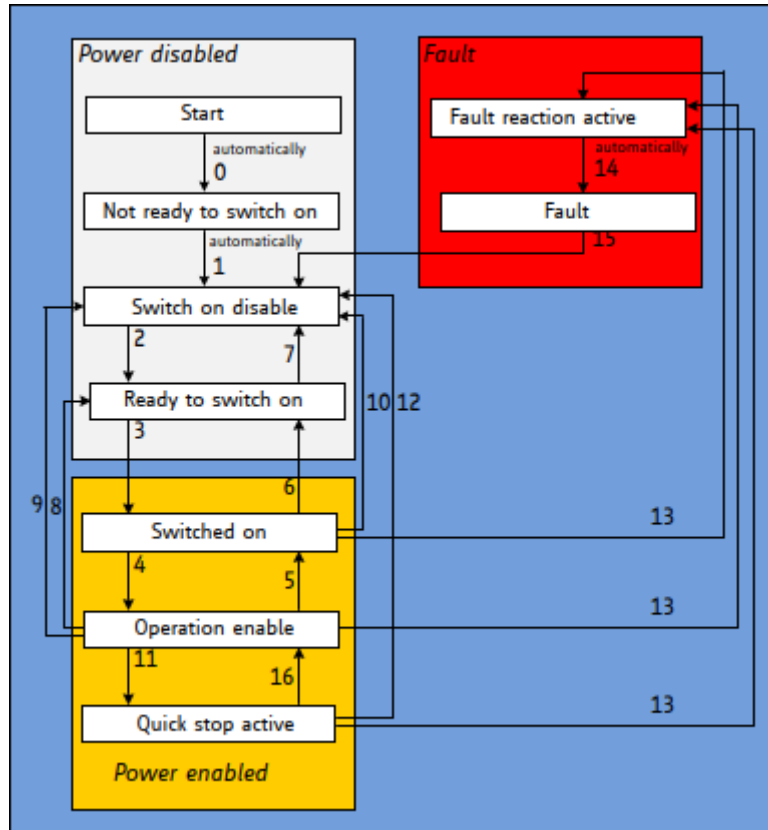
The index and subindex together determine the position of an object in the object dictionary, and the object length indicates the specific bit length of the object, represented in hexadecimal, that is:

Object length	bit
08h	8
10h	16
20h	32

- **Synchronization objects (SYNC)**

Synchronous object (SYNC) is a special mechanism that controls the coordination and synchronization between multiple nodes sending and receiving, used for synchronous transmission of PDO. The configuration process of the synchronous generator is as follows:





It is not recommended to use a synchronization cycle of less than 2ms for the drive.

- Synchronous generator

Drivers are both synchronous consumers and synchronous producers. The objects that support synchronization are the synchronization object COB-ID (1005h) and the synchronization cycle (1006h).

The second highest bit of the synchronization object COB-ID determines whether to activate the synchronization generator.

- Transmission framework for synchronous objects

Similar to the transmission of PDO, the transmission of synchronization objects follows the producer consumer model, where the synchronization producer sends out a synchronization frame, and all other nodes in the CAN network can receive the synchronization frame as consumers without feedback. Only one active synchronous generator is allowed in the same CAN network.

The transmission of synchronous PDO is closely related to synchronous frames.

For synchronous RPDO, once the PDO is received, it will be updated to the application in the next SYNC. For synchronous TPDO, it is divided into synchronous loop and synchronous non loop, as shown in the table below:

type		description
Synchronize	Synchronous non cyclic	The PDO transmission type is 0, and the content of the PDO mapping object has changed. It will be sent in the next SYNC.

TPDO	Synchronized cycle	The PDO transmission type is 1-240. As long as the SYNC specified in the transmission type is reached, regardless of whether the data has changed, the TPDO needs to be sent
------	--------------------	--

- Emergency Object Services (EMCY)**

When a CANopen node encounters an error, according to the standardization mechanism, the node will send an emergency message frame. The emergency message follows the producer consumer model, and after a node fault is sent, other nodes in the CAN network can choose to handle the fault. The driver only serves as an emergency message producer and does not process emergency messages from other nodes.

Objects related to emergency messages	Description
error register (1001h)	Reflecting the general error status of the node, each bit is classified according to its corresponding error
Booked in the wrong field (1003h)	Save recent errors

The emergency indication message is triggered by a fatal error within the device and has been sent to other devices with the highest priority by the relevant application devices. An error alarm signal suitable for interrupt types.

An emergency message consists of 8 bytes and has the following format:

Sending end → receiving end

COB-ID	Byte 0:1	Byte 2	Byte 3:7
0x080+Node-ID	Emergency error codes	Error Register (1001H)	Manufacturer designated area

Supported emergency error codes

The recent errors will be saved in the object dictionary (index 1003H) that is scheduled for the error field; Users can read this information through SDO; But if the drive loses power, these error messages will not be saved. The current error type is saved in the object dictionary error register (index 1001H).

The device can map internal errors to this status byte and quickly view the current error type.

The following table shows the definition of error register bits

bit	Error type
0	General error
1	current
2	voltage
3	temperature
4	communicate
5	Error specified in device protocol (402)
6	reserve
7	Manufacturer specified error

- Emergency error code

The emergency error code are divided into **Alarms** and **Warnings**

- **Lower 8 bits** represent the alarm code. Only one alarm code is displayed at a time.
- **Upper 8 bits** represent the warning code. Multiple warning codes can be displayed simultaneously.

For example for warning, if both **Positive Limit (2048)** and **Negative Limit (4096)** are triggered at the same time, the warning code will be **2048 + 4096 = 6144**.

Alarm Type	Function	Indicator Light Status	Fault Code (Decimal)	Description
Alarm	Motor Overcurrent	1 Green + 1 Red	10	Motor phase current overcurrent or driver fault
	Motor Phase Loss	1 Green + 2 Red	11	Motor not connected
	Motor Overcurrent	1 Green + 1 Red	12	Exceeded the set compensation limit
	Undervoltage	1 Green + 4 Red	13	Power input below 18V
	Overvoltage	1 Green + 3 Red	14	Power input above 60V
	Driver Overheating	1 Green + 5 Red	15	Driver overheating
	Driver Fault	1 Green + 5 Red	16	MOSFET driver voltage fault External power supply instability
	EEPROM Abnormal	1 Green + 5 Red	20	EEPROM data write error
	Overspeed Error	1 Green + 5 Red	24	Motor speed exceeds system maximum limit
	Position Deviation	1 Green + 5 Red	25	Position deviation exceeds set value
	Motor Overload	1 Green + 5 Red	26	Motor overload
	Bus Communication Abnormal	1 Green + 5 Red	50	Communication line disconnected or communication quality unstable while device is enabled
	Bus Communication Error	1 Green + 5 Red	100	Motor enters OP mode after the bus is disconnected in enabled state
Warning	EEPROM Data Read Error	Red light steady	256	EEPROM data read error
	Bus Voltage Instability	Red light steady	512	Bus voltage instability
	Emergency Stop (E-Stop)	Red light steady	1024	Emergency stop state
	Positive Limit	Red light steady	2048	On the positive limit or exceeding the positive soft limit
	Negative Limit	Red light steady	4096	On the negative limit or exceeding the negative soft limit
	Home Return Failure	Red light steady	8192	Home return failure

- Operation status

Status Function	Indicator Light Status	Communication Code	Description
Stopped	Green light flashing	2	Stopped
Running	Green light steady	3	Running
Enable Disconnected	Green light flashing	1	Enable Disconnected

### 3. CAN parameter description and settings

- SDO parameter list

The bus type closed-loop stepper driver is a standard EtherCAT slave device that follows the EtherCAT standard

protocol and can communicate with standard master stations that support this protocol.

The PC software interacts with the driver using the MODBUS protocol. The PC software can modify/read all

parameters, alarm information, and control the driver for trial operation.

- Configuration parameters

Index	Nom	Type	Accès	Défaut	Unité	Note
0x2064	Integrated Current	I16	RO	0	0.001A	
0x2065	Bus Voltage	U16	RO	0	1%V	
0x206C	Error Code	U16	RO	0		
0x206D	Run State	I16	RO	4		
0x207C	Locked_Rotor_Det	U16	RO	0		
0x2090	Analog Voltage	U16	RO		0.001V	Analog input value
0x20C8	Current loop Kpt	U16	RW			
0x20C9	Running direction	U16	RW			0: Default 1: Reverse running direction 2: Reverse encoder direction 3: Reverse running direction and encoder direction
0x20CE	Control mode	U16	WO	0		0: Pulse control 1: software control 5: Torque control 6: Position depression
0x20D0	L1e6	U16	RW	0		
0x20D1	R1e3	U16	RW	0		
0x20D5	Idle current	U16	RW			The stopping current in the percentage of operating current
0x20D7	Current loop KiS	U16	RW			

0x20D9	Motor mode setting	I16	RW			0: open loop 1: closed loop
0x20DB	LRD_Threshold_SPEED	U16	RW			
0x20DC	LRD_Threshold_GAIN	U16	RW			
0x20DD	LRD_Max_CNT	U16	RW			
0x20DE	Current loop Kp max	U16	RW			Maximum value of current loop Kp
0x20E0	Filter coefficient	U16	RW			The smaller the value, the smoother the motor runs, but the higher the delay.
0x20E1	Current Amplification Setting	U16	RW			
0x20E4	Current loop Ki Max	U16	RW			Maximum value of current loop Ki
0x20F1	Current command setting	I16	RW		0.001A	
0x20F2	Resolution setting	U32	RW		PPR	
0x20F5	Half Current Time Set	U16	RW		ms	Delay time for motor to enter stop state after stop running
0x20F6	Encoder resolution	U16	RW			Resolution = number of encoder lines x 4
0x20F7	In position range	U16	RW			
0x20F8	Max weak magnetic	U16	RW			
0x20FB	Speed loop Kp	U16	RW			
0x20FC	Speed loop Ki	U16	RW			
0x20FD	Speed loop Kd	U16	RW			
0x20FE	Maximum speed	I16	RW			
0x20FF	Position loop Kp	U16	RW			
0x2100	Position loop Ki	U16	RW			
0x2101	Position loop Kd	U16	RW			
0x2102	Position error threshold	U16	RW			Position deviation threshold, the value is the encoder resolution.
0x2103	Ref torque	I16	RW			
0x2108	Damping filter gain	U16	RW			
0x2109	Damping gain	U16	RW			
0x2110	Current filter gain	U16	RW			
0x2113	ARGain	U16	RW			
0x2114	ARAdj	U16	RW			
0x2115	APMGain	U16	RW			
0x2116	Break mode	U16	RW			
0x2117	Break vol	U16	RW			
0x2118	Break vol2	U16	RW			

0x2119	Break idle time	U16	RW			
0x211A	Break act delay	U16	RW			
0x211E	Check torque timer	I16	RW	10		
0x211F	Torque speed	I16	RW	500		
0x2127	Automatic detection parameters	U16	RW			In open-loop mode Automatically detect and update motor parameters 0: Manual settings 1: Automatic detection
0x2190	0x01	Digital input function 1	U16	RW		
	0x02	Digital input function 2	U16	RW		
	0x03	Digital input function 3	U16	RW		
	0x04	Digital input function 4	U16	RW		
	0x05	Digital input function 5	U16	RW		
	0x06	Digital input function 6	U16	RW		
	0x07	Digital input function 7	U16	RW		
	0x08	Digital input function 8	U16	RW		
0x21A4	0x01	Digital output function 1	U16	RW		
	0x02	Digital output function 2	U16	RW		
	0x03	Digital output function 3	U16	RW		
	0x04	Digital output function 4	U16	RW		
	0x05	Digital output function 5	U16	RW		
0x21AD	Digital input logic	U16	W			Digital inputs states
0x21AE	Digital output logic	U16	W			Digital output states
0x21E0	GXACCdata0	I32	RW			Global variable 1 value
0x21E2	GXACCdata1	I32	RW			Global variable 2 value
0x21E4	GXACCdata2	I32	RW			Global variable 3 value
0x21E6	GXACCdata3	I32	RW			Global variable 4 value
0x21E8	GXACCdata4	I32	RW			Global variable 5 value

▪ Motion parameters

Index	Nom	Type	Accès	Défaut	Unité	Note
0x603F	Error register	U16	RO	0x0		
0x6040	Control word	U16	RWW			
0x6041	Status word	U16	RO			
0x605A	Quick stop	I16	RW	0x2		
0x605B	Shutdown option code	I16	RW	0x0		Indicates what action is performed if there is a transition from operation enabled state to ready to switch on state

0x605C		Disable operation option code	I16	RW	0x1		indicates what action is performed if there is a transition from operation enabled state to switched on state
0x605E		Fault reaction option code	I16	RW	0x2		Indicates what action is performed when fault is detected in the power drive system
0x6060		Modes of operation	I8	RWW			1: pp 3: pv 6: Home 8: CSP
0x6061		Modes of operation display	I8	RO			
0x6064		Actual position	I32	RO		Pulse	
0x606C		Actual speed	I32	RO		RPS	
0x6071		Target torque	I16	RWW	0x0		
0x6074		Torque demand value	I16	RO	0x0		
0x6078		Current actual value	I16	RO	0x0		Torque mode only
0x607A		Target position	I32	RWW		Pulse	PP Mode 1 Target Position Command
0x607C		Origin offset	I32	RWW	0x0	Pulse	
0x607D	0x01	Minimal soft position limit	I32	RW	0x88CA6C00	Pulse	
	0x02	Maximal soft position limit	I32	RW	0x77359400	Pulse	
0x6081		Profile velocity	U32	RWW		RPS	PP mode 1 maximum speed
0x6083		Profile acceleration	U32	RWW		RPS <sup>2</sup>	PP, PV mode 1, 3 acceleration
0x6084		Profile deceleration	U32	RWW		RPS <sup>2</sup>	PP, PV modes 1, 3 deceleration
0x6085		Quick stop deceleration	U32	RWW		RPS <sup>2</sup>	Emergency stop deceleration (pp, pv, Home)
0x6087		Torque slope	U32	RWW	0x0		
0x6098		Homing method	I8	RWW	0x0		
0x6099	0x01	Homing speeds during search for switch	U32	RWW	0x0	RPS	

	0x02	Homing speeds during search for zero	U32	RWW	0x0	RPS	
0x609A		Homing acceleration	U32	RWW			
0x60FD		Digital inputs status	U32	RO	0x0		Inputs status
0x60FE	0x01	Digital outputs physical outputs	U32	RWW	0x0		Main station output signal control word
	0x02	Digital outputs Bitmask	U32	RWW	0x0		Control modes supported by the driver
0x60FF		Target velocity	I32	RWW		RPS	PV mode 3 Target velocity command
0x6502		Supported drive modes	U32	RO	37		Control modes supported by the driver

- **Essential functions**

After receiving the motion start command from the main station, the closed-loop stepper driver of this product will plan the trajectory according to the motion parameters sent by the main station; In asynchronous motion mode, the motion between each motor axis is asynchronous. This product's asynchronous motion modes include Protocol Position Mode (PP), Protocol Speed Mode (PV), and Origin Mode (HM).

- **Operatings modes**

Regardless of the control mode, data exchange between EtherCAT bus master and slave stations is achieved through an object dictionary. There are two types of data transmission methods: PDO and SDO. According to control needs, data transmission is divided into three levels based on real-time requirements and importance: mandatory>recommended>possible. "Must" means that in this mode, the corresponding object dictionary must be configured as PDO transmission mode. "Suggestion" indicates that in this mode, the corresponding object dictionary is recommended to be configured as PDO transmission mode to ensure real-time data and better control requirements; If the control requirements are not high, data transmission can also be carried out through SDO communication. "Can" indicates that in this mode, the corresponding object dictionary is generally transmitted through SDO communication and does not necessarily need to be configured as PDO. The object dictionaries associated with each control mode are shown in the table below.

Dictionary of objects associated with each control mode							
control model	Index+sub index	Name	data type	R/W	Unit	PDO mapping	SDO communication
PP mode (1)	607A-00h	Target Position	I32	RW	P	recommendation	feasible



	6081-00h	maximum speed	U32	RW	P	feasible	feasible
PV mode (3) PP mode (1) own	60FF-00h	target speed	I32	RW	P	recommendation	feasible
	6040-00h	control word	U16	RW	—	recommendation	feasible
	6083-00h	acceleration	I32	RW	P/S^2	feasible	feasible
	6084-00h	deceleration	U32	RW	P/S^2	feasible	feasible
HOME mode (6)	6040-00h	control word	U16	RW	—	recommendation	feasible
	6098-00h	Zero re- turn method	I8	RW	—	feasible	feasible
	6099-01h	Origin Fast	U32	RW	P/S	feasible	feasible
	6099-02h	Origin slow	U32	RW	P/S	feasible	feasible
	609A-00h	Origin acceleration	U32	RW	P/S^2	feasible	feasible
	607C-00h	Origin Offset	U32	RW	P	feasible	feasible
PP, PV & HOME mode	6041-00h	status word	U16	RO	—	recommendation	feasible
	6064-00h	ACTUAL POSITION	I32	RO	P	recommendation	feasible
	606C-00h	Actual Speed	I32	RO	P/S	feasible	feasible
All mode	60FD-00h	digital input	U32	RO	—	recommendation	feasible
	603F-00h	Latest error codes	U16	RO	P	recommendation	feasible
Other associated parameters	6060-00h	MODE	I8	RW	—	feasible	feasible
	60B0-00h	Position off- set	I32	RW	—	feasible	feasible
	6082-00h	Jumping speed	U32	RW	P/S	feasible	feasible
	6085-00h	Emergency stop decel- eration	U32	RW	P/S^2	feasible	feasible
	6061-00h	Operation mode display	I8	RO	—	feasible	feasible

No matter which control mode is used to achieve driving control of the actuator, it cannot be separated from the reading and writing of the control word 6040h and the status word 6041h object dictionaries. The master and slave stations use these two object dictionaries as media to issue instructions and monitor the status. The following focuses on the definitions of each bit in these two object dictionaries.

▪ Control and status word

The definition of control word (6040h) is shown in the table below. The left half of the table describes bits 4 to 6 and bit 8, which depend on the operating mode and mainly control the execution or stopping of each mode; The right half of the table describes bit0~3 and bit7, which together manage the state transition changes of the 402 state machine to meet complex and diverse control requirements. The definition of status word (6041h) is shown in the definition table of status word (6041h) bits. Bit0~bit7 mainly displays the transition status of the DS402 state machine, while bit8~bit15 mainly displays the motion execution or stop status in various control modes. The typical state transitions that enable are as follows:

Initial (00h) - Power on (06h) - Start (07h) - Enable (0fh) - Execute run or pause (depending on the operating mode, combined with bit4-6 and bit8)

Issue relevant control instructions. The state transitions that trigger operation control in each control mode are shown in the state transition table for each mode control operation.

Definition of control word (6040h) bits												
Mode /bit	15 ~9	8	6	5	4	7	3	2	1	0	Typical value	Action result
all	-	suspend	Depending on the operating mode			Error reset	Allow	Quick Stop	Voltage Output	firing		
PPmode1	-	Deceleration Stop	Abs/ref	Immediately trigger	New Location point	0	0	1	1	1	07h	firing
PVmode3	-	Deceleration Stop	Non	Non	Non	0	0(x)	0	1	0(x)	02h	Quick stop
HMmode6	-	Deceleration Stop	non	Non	Start movement	0	1	1	1	1	0fh	Able

Additional explanation for other positions:

Bit 2 quick stop trigger logic is 0 effective, please distinguish it from other triggered logic.

Bit 7 error reset trigger logic is effective on the rising edge.

Bit 5 immediately triggers the triggering logic, which is effective on the rising edge.

Status word (6041h) bit definition								
mode/low8bit	7	6	5	4	3	2	1	0
All	reserve	Switch on disabled	Quick Stop	Voltage enabled	fault	Operation enabled	Switched on	ready to switch on

mode/ high8bit	15	14	13	12	10	8	11	9
All	Depending on the operating mode						Limit effective	remote
PPmode1	Triggering response	Parameter has 0	No	New Location Point response	Position Reach	abnormal stop	Set when hardware limit is valid	0 below PreOP
PVmode3	no	Parameter has 0	no	Speed is 0	Speed reached	Quick Stop		
HM mode 6	Triggering response	Parameter has 0	Origin error	Origin completed	Position Reach	abnormal stop		

Additional explanation for other positions:

When the drive is powered on, bit 4 will be set.

Bit 5 quickly stops activation and is only effective under logic 0, which is opposite to the logic of other bits.

Bit 9 is remote, displaying the communication status machine status, which is 0 below ProOP. At this time, the command of the control word (6040h) will not be executed.

Bit 11 limit is set only when the hardware limit is valid.

Bit 8 stops abnormally and is generally effective when triggered by hardware limit, deceleration stop, and fast stop.

Bit 12 follows the master station, and in CSP, if the driver is not enabled or no longer follows the master station's instructions, this position is 0.

State transition of each mode control operation										
	Step	0	1	2	3	4	5	6	7	8
Mode	Action	Preparatory work	initial	Power on	firing	enable	start-up	Displacement	cease	Fault
PP mode 1	6040	Establish communication OP status and set motion parameters	00h	06h	07h	0Fh	-	2Fh->3Fh	10Fh	-
	6041		250h	231h	233h	8237h	1237h	1637h->1237h	1737h	1238h
PV mode 3	6040	Establish OP status and set motion parameters	00h	06h	07h	0Fh	Run immediately after enabling	Change speed is sufficient	10Fh	-
	6041		250h	231h	233h	1637h	1637h	1637h	173	1638h

HM mode 6	6040	Establish OP status and set motion parameters	00h	06h	07h	0fh	1fh	invalid	10Fh	-
	6041		250h	231h	233h	833	237h	237h	737h	238h

Additional explanation for other positions:

When changing the position in PP mode, it is necessary to give the bit5 rising edge of the control word in order to start a new position movement.

- **Default mapping PDO**

For the eds file version 4.0, find the default PDO mapping below:

PDO	Mapping
RPDO1	[0x6040,0x00] Controlword (Unsigned16)
	[0x6060,0x00] Modes of operation (Integer8)
	[0x6098,0x00] Homing_method (Integer8)
	[0x607C,0x00] Home_offset (Integer32)
RPDO2	[0x607A,0x00] Target position (Integer32)
	[0x6081,0x00] Profile velocity (Unsigned32)
RPDO3	[0x60FF,0x00] Target velocity (Integer32)
	[0x6085,0x00] Quick stop deceleration (Unsigned32)
RPDO4	[0x6083,0x00] Profile acceleration (Unsigned32)
	[0x6084,0x00] Profile deceleration (Unsigned32)
TPDO1	[0x6041,0x00] Statusword (Unsigned16)
	[0x603F,0x00] Error_code (Unsigned16)
	[0x6061,0x00] Modes of operation display (Integer8)
TPDO2	[0x6064,0x00] Position actual value (Integer32)
	[0x606C,0x00] Velocity actual value (Integer32)
TPDO3	[0x207C,0x00] Locked_Rotor_Det (Unsigned16)
	[0x60FD,0x00] Digital_inputs (Unsigned32)
TPDO4	<no mapped entries>