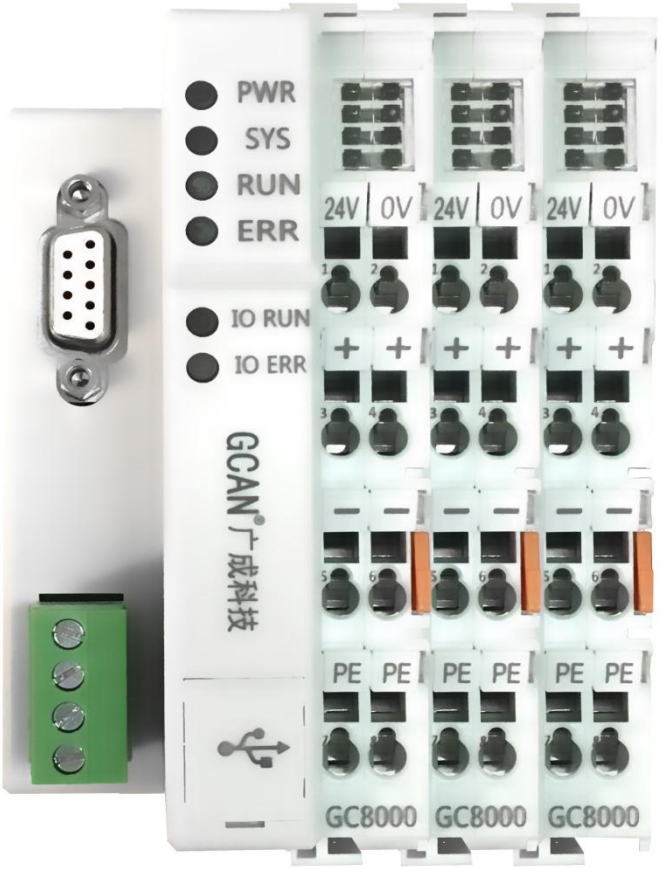
GCAN-IO-8000

CANopen bus coupler

User manual



Document version: V3.25 (2019/03/20)

revise history

Version	Date	The reason
V1.00	2015/10/16	Create document
V2.01	2015/12/20	Correct equipment working parameters
V3.01	2017/11/22	Add some parameters
V3.02	2018/01/22	Add communication protocol section
V3.03	2018/03/22	Correct equipment working parameters
V3.25	2019/3/20	Modify selection table

catalog

1. Function introduction....4
1.1 Functional Overview....4
1.2 Performance characteristics....4
1.3 Typical applications....5
2. Equipment installation and use....5
2.1 Module appearance and dimensions....5
Figure 2.1 Appearance of GCAN-IO-8000 CANopen bus coupler....5
2.1 Module fixing....5
2.2 Wiring method....7
2.3 System status indicator....9
3. Communication connection....11
3.1 Serial port connection....11
3.2 CAN connection....11
Table 3.1 Baud rate and maximum bus length reference table....12
3.3 CAN bus termination resistance....12
3.4 Setting of CAN baud rate and node number....12
4. Communication protocol....15
4.1 NMT command....15
Figure 4.3 NMT command monitoring interface....17
4.2 Equipped with GC-1008 module....17
4.3 Equipped with GC-2008 module....18
4.4 Equipped with GC-3804 module....19
4.5 Equipped with multiple sets of modules at the same time....20
5. Technical specifications....23
6. GC series module selection table....24
Appendix A: CAN2.0A protocol frame format....26
Appendix B: Introduction to CANopen Protocol....27
B.1 Explanation of related terms and writing rules....27
B.2 Predefined CAN identifier....28
B. 3 CANopen object dictionary....29
B.4 CANopen communication....29
B.5 CANopen network configuration....34
Sales and Service....35

1. Function introduction

1.1 Functional Overview

GCAN-IO-8000 CANopen bus coupler can be used to connect CAN bus system and distributed bus terminal modules. These terminal modules can be expanded in a modular manner. A complete node consists of a bus coupler, 1-32 any number of terminal modules and a terminal terminal module. Adopt GCAN-IO-8000 bus coupler, through GC-bus expansion technology, can establish I/O connection very conveniently, can connect up to 32 input/output terminal modules.

GCAN-IO-8000 CANopen bus coupler adopts CAN bus protocol that conforms to ISO 11898 standard. The GCAN-IO-8000 bus coupler not only supports all types of CANopen communication, but can also be easily applied to manufacturer-specific CAN bus environments. In addition, the firmware can be upgraded through the configuration interface.

GCAN-IO-8000 bus coupler can connect all bus terminal modules. As far as the user is concerned, the processing of analog input/output signals is no different from the processing of other types of signals. The information in the process image area of the controller is displayed in byte array format. According to different models, the analog bus terminal module register contains the temperature range, gain value and linearized characteristic curve.

GCAN-IO-8000 bus coupler supports automatic configuration, you do not need to set parameters on the PC. The CANopen baud rate of the GCAN-IO-8000 bus coupler can be configured via the RS-232 interface.

1.2 Performance characteristics

- CANopen baud rate supports 1000k, 500k, 250k, 125k, 100k, 50k, 10k;
- PDO mode supports synchronization, loop, event-driven, and polling;
- The number of bus terminal modules is 32;
- Send 12 PDO (CANopen), receive 12 PDO (CANopen);
- The configuration mode is automatic configuration;
- The CAN bus interface is an open 4-pin terminal;
- The power supply adopts 24V DC (-15%/+20%);
- The input current is 70mA+ (total GC-bus current), the maximum is 2.5A;
- Starting current: about 2.5 times the continuous current;
- Power supply: Max 24V DC/Max 10A;
- The electrical isolation is 1500 Vrms;
- Working temperature range: -40℃~+85℃;
- Dimensions: length 100mm * width 69mm * height 48mm.

1.3 Typical applications

Connect with the distributed bus terminal module to form a complete control node; Perform data collection and data transmission with CANopen protocol.

2. Equipment installation and use

This chapter will explain in detail the installation method, wiring method, the meaning of the indicator light and the meaning of the interface of GCAN-IO-8000 CANopen bus coupler.

2.1 Module appearance and dimensions

The appearance of GCAN-IO-8000 is shown in Figure 2.1. GCAN-IO-8000 CANopen bus coupler includes 2 communication interfaces, 1 controller programming interface, 1 group controller power interface, 2 groups I/O power interface, 2 groups shielded wire interface. Among them, the communication interface includes a CAN bus interface and an RS232 interface.

Figure 2.1 Appearance of GCAN-IO-8000 CANopen bus coupler

2.1 Module fixing

The installation method of GCAN-IO-8000 CANopen bus coupler is shown in figure 2.2, you need to use a flat screwdriver for auxiliary installation.

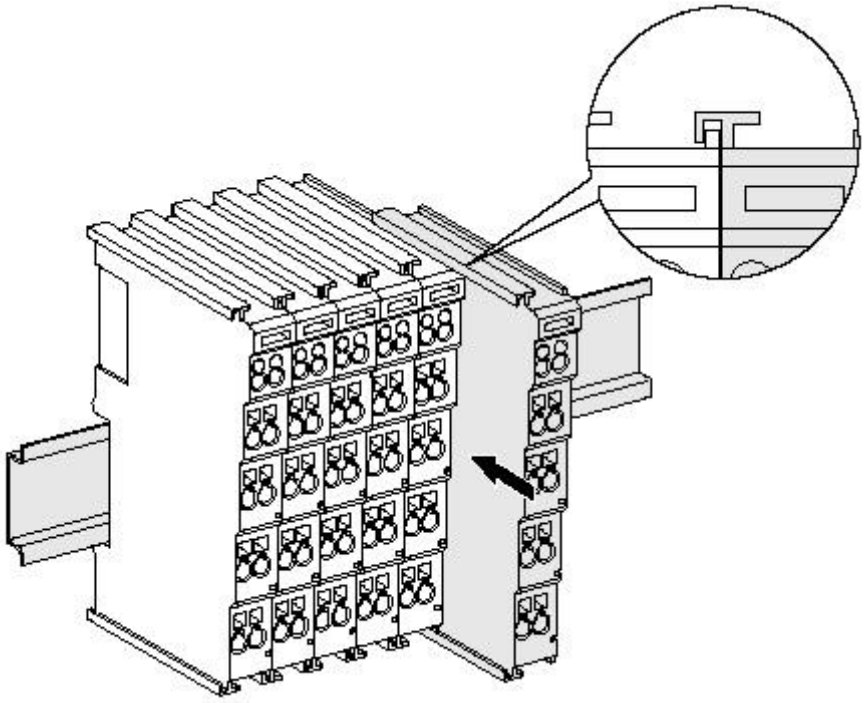


Figure 2.2 GCAN-IO-8000 module installation

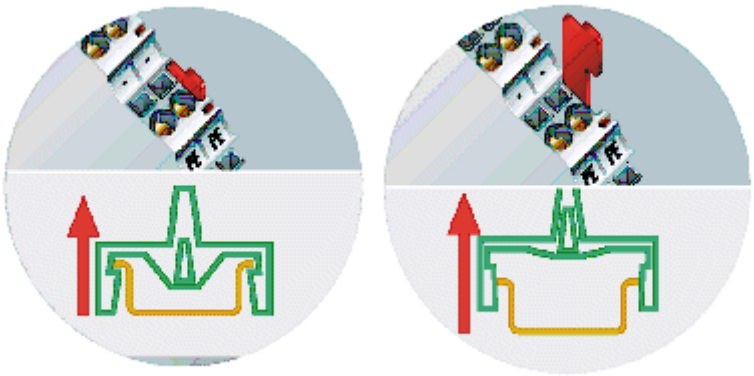


Figure 2.3 GCAN-IO-8000 module self-locking mechanism

Please install the GCAN-IO-8000 CANopen bus coupler on the guide rail as shown in Figure 2.3 until the latch snaps and makes a “click” sound. GCAN-IO-8000 CANopen bus coupler has a self-locking mechanism, which can effectively prevent the device from falling. As shown in Figure 2.3, you can release the self-locking mechanism by pulling out the orange label.

GCAN-IO-8000 CANopen bus coupler can connect up to 32 distributed bus terminal modules. When inserting the bus terminal module, be sure to follow the groove and insert it on the right side of the existing module in sequence until the latch snaps and makes a “click” sound. At the far right end of the entire node, you need to install a terminal module. The terminal can guarantee the data transmission and power supply of GC-Bus.

When you assemble the nodes correctly, there will be no obvious gaps between the

terminal modules. If the modules are not assembled correctly, the entire node will not operate normally.

2.2 Wiring method

As shown in Figure 2.4, use a flat-blade screwdriver to insert into the square hole, press the upper edge of the metal sheet in the square hole, and press firmly in the direction of the round hole. Then insert the cable into the circular hole. After plugging in, pull out the screwdriver, and the cable can be firmly locked in the circular hole

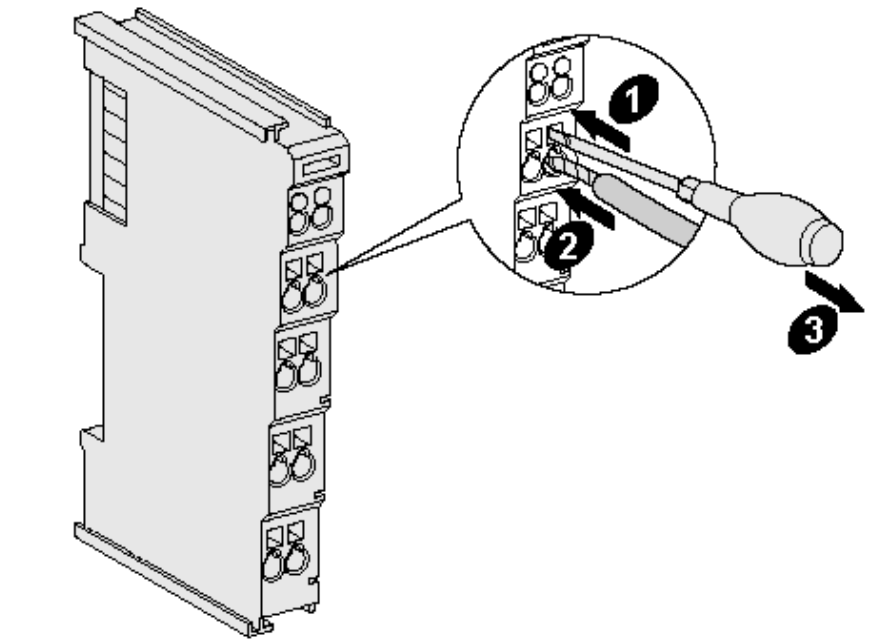


Figure 2.4 Wiring of GCAN-IO-8000 module power supply

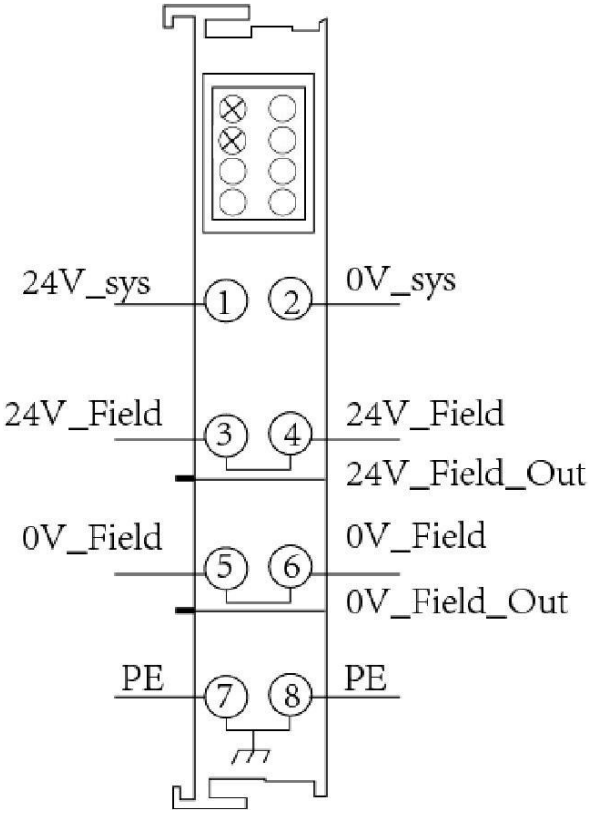


Figure 2.5 GCAN-IO-8000 module power terminal block

The power terminal block of GCAN-IO-8000 CANopen bus coupler is shown in Figure 2.5. The GCAN-IO-8000 CANopen bus coupler contains 8 terminals, and the corresponding serial numbers and meanings of each terminal are shown in Table 2.1. Please note that between terminal 3 and terminal 4, between terminal 5 and terminal 6, and between terminal 7 and terminal 8 are connected inside the module.

Terminal	Serial number	meaning
24V	1	Power 24V input
0V	2	Power GND
+	3	IO power supply is positive
+	4	IO power supply is positive
-	5	IO power negative
-	6	IO power negative
PE	7	shield
PE	8	shield

Table 2.1 Definition of GCAN-IO-8000 module power terminal

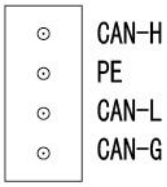


Figure 2.6 GCAN-IO-8000 module CAN bus terminal block

The CAN bus terminal block of GCAN-IO-8000 CANopen bus coupler is shown in Figure 2.6. The CAN bus terminal block of GCAN-IO-8000 contains 4 terminals, and the corresponding serial number and meaning of each terminal are shown in Table 2.2.

Terminal	Serial number	meaning
CAN-H	1	CANopen high
PE	2	Shielded wire
CAN-L	3	CANopen low
CAN-G	4	CANopen ground

Table 2.2 Definition of CAN bus terminal of GCAN-IO-8000 module

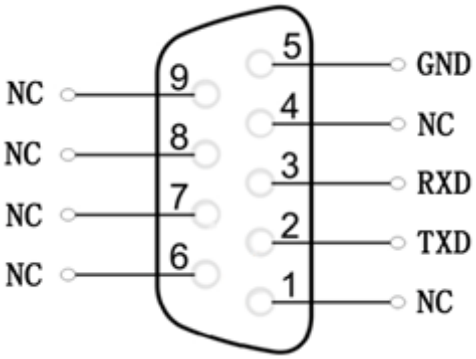


Figure 2.7 GCAN-IO-8000 module RS-232 interface definition

The definition of RS-232 interface of GCAN-IO-8000 CANopen bus coupler is shown in Figure 2.7. GCAN-IO-8000's RS-232 interface only defines three signal lines, namely RXD, TXD and GND.

2.3 System status indicator

GCAN-IO-8000 CANopen bus coupler has two sets of status indicators. The left area contains 6 circular status indicators, and the right area contains 2 small power indicators. The specific indication function of the indicator light is shown in Table 2.3. When the indicator lights are in different states, the state of the GCAN-IO-8000 module is shown in Table 2.4.

Indicator light	colour	Indication status
PWR	green	Power indicator
SYS	green	System instructions

RUN	green	Operating instructions
ERR	green	Error indication
IO RUN	green	Internal bus operation instructions
IO ERR	green	Internal bus error indication
Position 1 on the right	green	Power indicator
Position 3 on the right	green	Internal bus power indication

Table 2.3 GCAN-IO-8000 module indicator

Indicator light	status	Indication status
PWR	Always on	Power supply is normal
	not bright	Abnormal power supply
SYS	flicker	The device is initialized and enters the working state
	not bright	Device initialization failed
RUN	flicker	The device is operating normally
	not bright	Device operation stopped
ERR	Always on	system error
	not bright	No errors in the system
IO RUN	flicker	The internal bus is operating normally
	not bright	Internal bus stop
IO ERR	Always on	Internal bus operation error
	not bright	No error occurred during internal bus operation
Position 1 on the right	Always on	Normal power supply on the terminal side
	not bright	Abnormal power supply on the terminal side
Position 3 on the right	Always on	The internal bus power supply of the terminal is normal
	not bright	Abnormal internal bus power supply of the terminal

Table 2.4 GCAN-IO-8000 module indicator status

3. Communication connection

3.1 Serial port connection

GCAN-IO-8000 CANopen bus coupler uses standard serial port level (RS232: ±3~15V), so the module can be directly connected to devices with RS232 interface.
The baud rate of the RS232 interface of the GCAN-IO-8000 CANopen bus coupler cannot be modified and is fixed at 19200bps.

3.2 CAN connection

GCAN-IO-8000 module is connected to the CAN bus as described in 2.2. Connect CAN_H to CAN_H and CAN_L to CAN_L to establish communication.
The CAN-bus network adopts a straight-line topology structure, and the two furthest terminals of the bus need to install 120Ω terminal resistance; if the number of nodes is greater than 2, the intermediate nodes do not need to install 120Ω terminal resistance. For branch connections, the length should not exceed 3 meters. The connection of CAN-bus bus is shown in Figure 3.1.

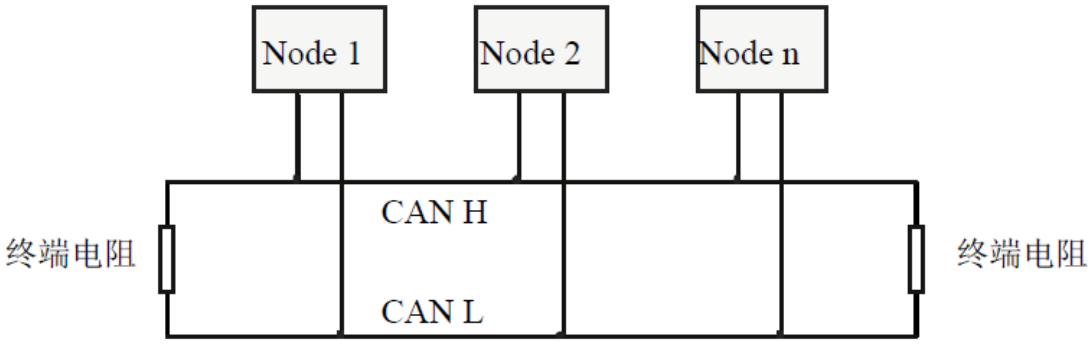


Figure 3.1 Topology of CAN-bus network

Please note: CAN-bus cable can use ordinary twisted pair and shielded twisted pair. The theoretical maximum communication distance mainly depends on the bus baud rate. For the relationship between the maximum bus length and baud rate, see Table 3.1. If the communication distance exceeds 1km, the cross-sectional area of the line should be greater than Φ1.0mm2, the specific specifications should be determined according to the distance, and the conventional is to increase appropriately as the distance increases.

Baud rate	Bus length
1 Mbit/s	40m
500 kbit/s	110m
250 kbit/s	240m
125 kbit/s	500m
50 kbit/s	1.3km
20 kbit/s	3.3km
10 kbit/s	6.6km
5 kbit/s	13km

Table 3.1 Baud rate and maximum bus length reference table

3.3 CAN bus termination resistance

In order to enhance the reliability of CAN communication and eliminate CAN bus terminal signal reflection interference, the two farthest endpoints of the CAN bus network usually need to add terminal matching resistors, as shown in Figure 3.2. The value of the termination matching resistance is determined by the characteristic impedance of the transmission cable. For example, the characteristic impedance of the twisted pair is 120Ω , then the two endpoints on the bus should also integrate 120Ω termination resistors. If other nodes on the network use different transceivers, the termination resistance must be calculated separately.

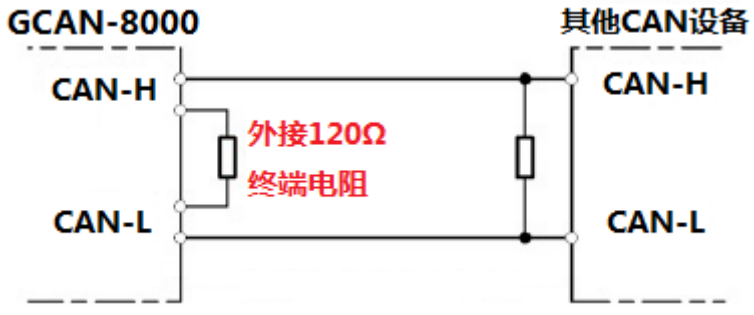


Figure 3.2 GCAN-IO-8000 connected with other CAN node devices

Please note: The 120Ω termination resistor is not integrated inside the GCAN-IO-8000 module. If the number of nodes is greater than 2, the intermediate node does not need to install a 120Ω termination resistor. When needed, connect CAN_H and CAN_L at both ends of the resistor, as shown in Figure 3.2.

3.4 Setting of CAN baud rate and node number

You can modify the CAN bus communication baud rate of the GCAN-IO-8000 module through the RS232 interface. The specific process is as follows: first establish

a serial port connection, the baud rate is 19200bps, the data bit and stop bit are configured as none, 8, 1, send a "help" command or "?" command through the RS232 interface, and then the RS232 interface will return to the configuration information. **Please note that when sending the command, please add a carriage return at the end or select the "send new line" of the software.** For display and transmission, please do not select "hexadecimal display" or "hexadecimal transmission". You can send "setbaud=500000" to modify the baud rate of the GCAN-IO-8000 module to 500kbps. At the same time, you can send "getbaud" to get the CAN baud rate of the machine, and the unit of the reply value is bps. The detailed baud rate correspondence is shown in Table 3.2. The settings when sending the request command are shown in Figure 3.3.

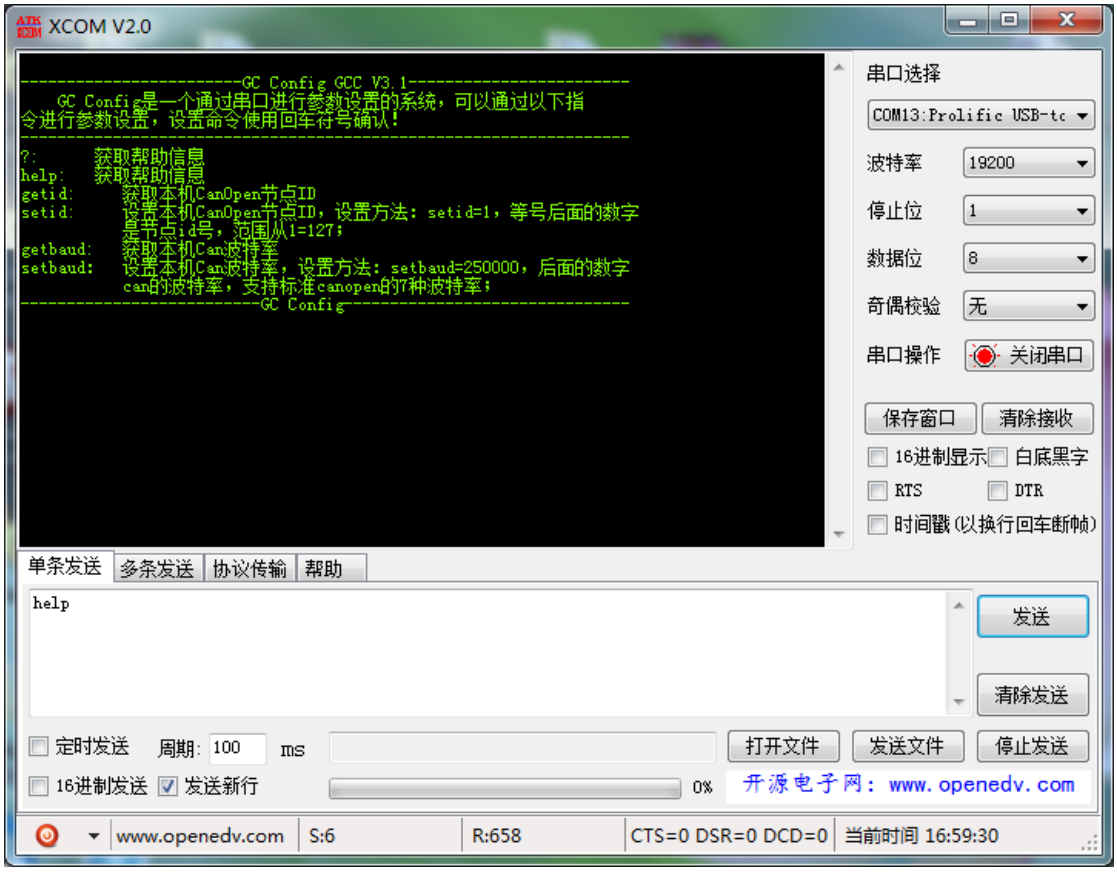


Figure 3.3 Settings when sending a request command

baud setting value	Corresponding baud rate (kbps)
setbaud=1000000	1000
setbaud=500000	500
setbaud=250000	250
setbaud=125000	125
setbaud=100000	100
setbaud=50000	50
setbaud=10000	10

Table 3.2 Baud setting value and baud rate comparison table

The method of setting and obtaining the CANopen node ID is similar to the baud rate.

You can modify the CAN bus node number of the GCAN-IO-8000 module through the RS232 interface. You can send "setid=3" to modify the CANopen node ID of GCAN-IO-8000 module to 3. At the same time, you can send "getid" to get the CANopen node ID of this machine. The node number can be set arbitrarily within 1-127.

4. Communication protocol

GCAN-IO-8000 implements CANopen communication protocol and is a CANopen slave device. The GCAN-IO-8000 module uses PDO (Process Data Object) to collect or output digital signals.

When equipped with GC-3804 or GC-1008 module, GCAN-IO-8000 will send out TPDO data, typical frame ID such as 0x181, 0x281, etc. When equipped with GC-2008 module, GCAN-IO-8000 will receive RPDO data, typical frame ID such as 0x203, 0x303, etc.

In this chapter, Guangcheng Technology USBCAN-II Pro module and ECANTools software can be used to receive and send CAN bus data. The CANopen master station function provided with the software can help debug the CANopen slave station, which is very convenient and practical.

Using the USBCAN bus analyzer of Guangcheng Technology can simulate the CAN bus communication device and conduct the communication test of the GCAN-IO-8000 module. The USBCAN bus analyzer is an intuitive CAN bus debugging and analysis tool. Using this device can monitor and simulate CAN data transmission and reception through a computer. It is an essential tool for engineers engaged in the CAN bus industry. You are welcome to purchase through the contact information on the last page of this manual.

4.1 NMT command

The GCAN-IO-8000 module meets the standard CANopen Cia301 protocol and is a standard CANopen slave device. **After GCAN-IO-8000 is started, it will actively send a frame command to the master station, the frame ID is 0x700+Node ID, and the frame data is 0x7F.**

For example: Set the Node ID of GCAN-IO-8000 to 1 through the DIP switch, then USBCAN-II Pro as the master device can receive a start command, the frame ID is 0x701, and the frame data is 0x7F. As shown in Figure 4.1, using ECANTools software can receive this data.

Frame ID（HEX）	DLC	Frame data (HEX)							
701	1	7F	--	--	--	--	--	--	--

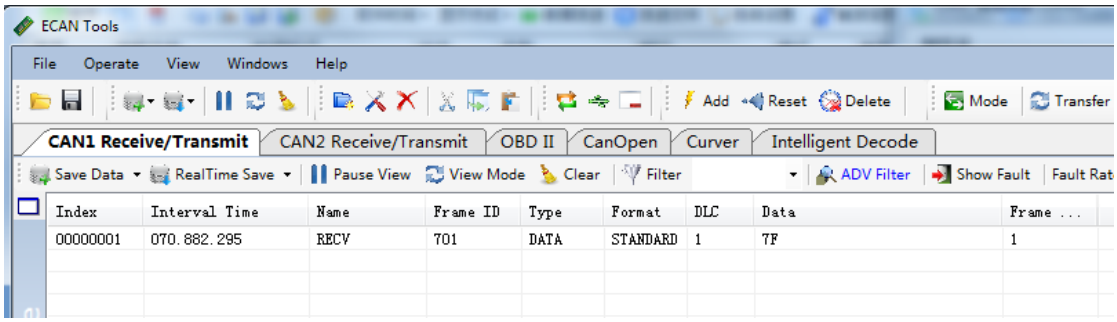


Figure 4.1 Start command monitoring interface

The GCAN-IO-8000 module receives the operation command issued by the master station, the frame ID is 0x000, the DLC is 2, the first byte of the frame data is the command symbol, and the second byte is the node number (00 is all nodes) .

For example: The node ID of GCAN-IO-8000 is 1, and the command GCAN-IO-8000 is to enter the operation state (01), then the NMT command frame ID is 0x000, and the frame data is 0x01,0x01. This command can also be issued by the CANopenMaster plug-in of the ECANTools software. For detailed instructions of NMT, please refer to Appendix B.4 CANopen Communication.

Frame ID (HEX)	DLC	Frame data (HEX)							
000	2	Command	Node address						
		01	01	--	--	--	--	--	--

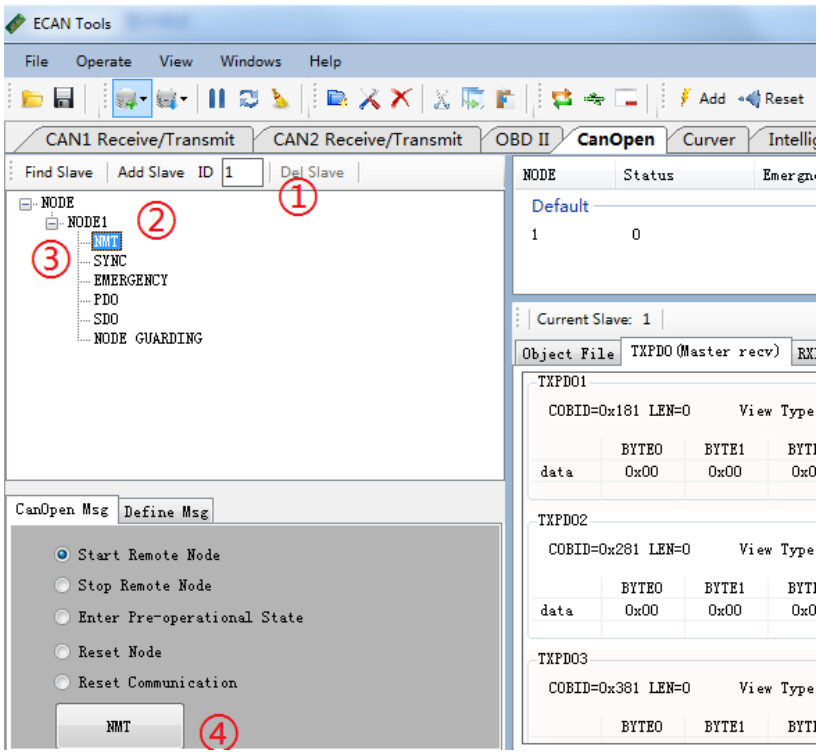


Figure 4.2 NMT command sending interface

Use the ECANTools software to command GCAN-IO-8000 to enter the operation state: ① fill the slave node number into the text box behind the ID, ② click the add slave button, ③ open the + sign on the left of the node, and click " NMT", ④ Select "Start Remote Node" (default) and click the "NMT" button below. As shown in Figure 4.3, the NMT command can be viewed in the ECANTools monitoring interface after sending.

After receiving the NMT command, GCAN-IO-8000 will start broadcasting PDO data and issue a heartbeat command with frame ID 0x701 and frame data 0x05, indicating that GCAN-IO-8000 has entered the operating state.

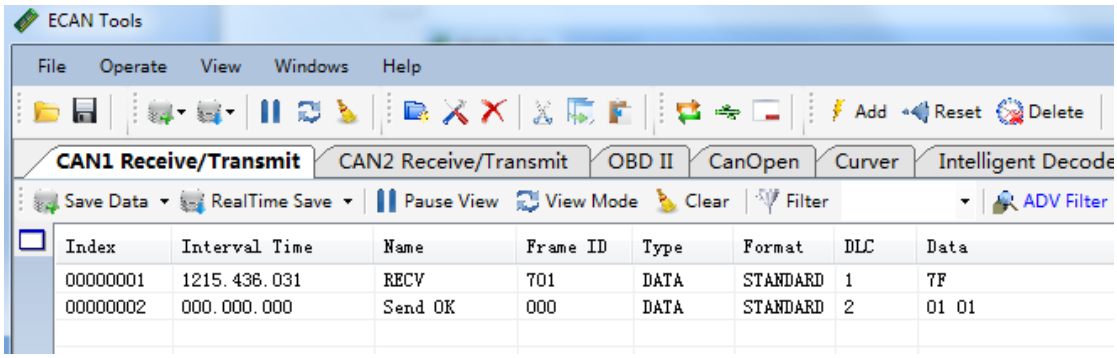


Figure 4.3 NMT command monitoring interface

4.2 Equipped with GC-1008 module

The state of the digital input is represented by a byte, channel 8 is in the high bit, and channel 1 is in the low bit.

For example, the node number of the GCAN-IO-8000 module is set to 1. The state of

channel 8 and channel 4 is 1, and the other states are all 0, then the DI status data displayed on one end of the CAN bus is 88. The frame ID sent is 0x181, the data length (DLC) is 8, the frame data is 0x88, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00. Please note that when only one GCAN-1008 module is inserted, only the first byte in the frame data is valid. The following table lists two common DI states and their corresponding state data.

DI status								
Number of channels	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
status	**1**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
Data displayed on the CAN bus	**88**							

DI status								
Number of channels	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
status	**0**	**1**	**0**	**1**	**1**	**0**	**1**	**0**
Data displayed on the CAN bus	**5A**							

When only one GCAN-1008 module is inserted, the TPDO data frame ID issued by the GCAN-IO-8000 module is 0x180+node ID (Node ID), the data length is 8, and the first byte of the frame data is the digital input status of the module .

4.3 Equipped with GC-2008 module

The state of the digital output is represented by a byte, channel 8 is in the high bit, and channel 1 is in the low bit.

For example, the node number of the GCAN-IO-8000 module is set to 1. Need to set the state of channel 8 and channel 4 to 1, and set all other states to 0, then the CAN

bus DO status data to be sent is 88. The frame ID to be sent to GCAN-IO-8000 is 0x201, the data length (DLC) is 8, and the frame data is 0x88, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00. Please note that only the first byte in the frame data is valid. The following table lists two common DO states and their corresponding state data.

DO status								
Number of channels	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
status	**1**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
Data displayed on the CAN bus	**88**							

DO 状态								
Number of channels	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**
status	**0**	**1**	**0**	**1**	**1**	**0**	**1**	**0**
Data displayed on the CAN bus	**5A**							

When receiving the RPDO data, the GCAN-IO-8000 module needs to ensure that the frame ID is 0x200 + node ID (Node ID), the data length is 8, and the first byte of the frame data is the digital output status that needs to be set.

4.4 Equipped with GC-3804 module

The temperature status of each channel is represented by two bytes, and the four channels have a total of eight bytes.

Among them, the two bytes representing the temperature status, the first byte is the low bit, the data of the byte needs to be converted to decimal and multiplied by 0.1; the second byte is the high bit, the data of the byte needs to be converted Multiply by 25.6 after decimal. Finally, the two values are added together to obtain the final temperature value in degrees Celsius.

For example, the node number of the GCAN-IO-8000 module is set to 1. The temperatures of the four channels are 25.6 degrees, 25.5 degrees, 20 degrees, and 30 degrees, respectively. The frame ID sent is 0x181, the data length (DLC) is 8, the frame data is 0x00, 0x01, 0xFF, 0x00, 0xC8, 0x00, 0x2C, 0x01. The following table lists two possible CAN data and their corresponding temperature values.

Correspondence between GC-3804 temperature and CAN data		
Data displayed on the CAN bus	**Low byte C8**	**High byte 00**
coefficient	**200（0xC8）x0.1**	**0（0x00）x25.6**
Temperature value	**20℃**	

Correspondence between GC-3804 temperature and CAN data		
Data displayed on the CAN bus	**Low byte 2C**	**High byte 01**
coefficient	**44（0x2C）x0.1**	**1（0x01）x25.6**
Temperature value	**30℃**	

When the GCAN-IO-8000 module sends out TPDO data, the frame ID is 0x180+node ID (Node ID), and the data length is 8. If PT100 is not connected, the CAN data of the corresponding channel will be displayed as FF 7F.

4.5 Equipped with multiple sets of modules at the same time

If GCAN-IO-8000 is equipped with multiple sets of GC-1008 modules at the same time, then we will number them according to their distance from GCAN-IO-8000 from near to far, and the nearest one is No. 1. The TPDO data sent by the GCAN-IO-8000 coupler will be sent according to the following table. For example, when the GCAN-IO-8000 node number is 3 and it is equipped with 9 GC-1008 modules at the same time, you will receive two sets of data with frame IDs 0x183 and 0x283. Among them, the eight data bytes with the frame ID of 0x183 correspond to the GC-1008 module No. 1-8 in turn; the one data byte with the frame ID of 0x283 corresponds to the GC-1008 module No. 9.

Frame ID	Frame data							
0x180+Node ID	1	2	3	4	5	6	7	8
0x280+Node ID	9	10	11	12	13	14	15	16
0x380+Node ID	17	18	19	20	21	22	23	24
0x480+Node ID	25	26	27	28	29	30	31	32

Table 4.1 Correspondence between multiple sets of GC-1008 modules and CAN data

If GCAN-IO-8000 is equipped with multiple sets of GC-2008 modules at the same time, then we number them according to their distance from GCAN-IO-8000, from near to far, and the nearest one is No. 1. The RPDO data received by the GCAN-IO-8000 coupler needs to be sent according to the following table. For example, when the GCAN-IO-8000 node number is 3 and 9 GC-2008 modules are installed at the same time, you need to send two sets of data with frame IDs 0x203 and 0x303 to control all 9 modules. Among them, the eight data bytes with the frame ID of 0x203 correspond to the GC-2008 module No. 1-8 in turn; one data byte with the frame ID of 0x303 corresponds to the GC-2008 module No. 9.

Frame ID	Frame data							
0x200+Node ID	1	2	3	4	5	6	7	8
0x300+Node ID	9	10	11	12	13	14	15	16
0x400+Node ID	17	18	19	20	21	22	23	24
0x500+Node ID	25	26	27	28	29	30	31	32

Table 4.2 Correspondence between multiple sets of GC-2008 modules and CAN data

If GCAN-IO-8000 is equipped with multiple sets of GC-3804 modules at the same time, then we number them according to their distance from GCAN-IO-8000, from near to far, and the closest is No. 1. The TPDO data sent by the GCAN-IO-8000 coupler will be sent according to the following table. For example, when the GCAN-IO-8000 node number is 3 and four GC-3804 modules are installed at the same time, you will receive four sets of data with a frame ID of 0x183. Among them, the eight data bytes with the frame ID of 0x183 correspond to the four channels of the GC-3804 module in sequence; the eight data bytes with the frame ID of 0x283 correspond to the four channels of the GC-3804 module in sequence.

Frame ID	Frame data
0x180+Node ID	number 1 4 channels of GC-3804
0x280+Node ID	number 2 4 channels of GC-3804
0x380+Node ID	number 3 4 channels of GC-3804
0x480+Node ID	number 4 4 channels of GC-3804

Table 4.3 Correspondence between multiple sets of GC-3804 modules and CAN data

If GCAN-IO-8000 is equipped with GC-1008 module and GC-3804 module at the same time, then we number them according to their distance from GCAN-IO-8000, from near to far, and the nearest one is No. 1. The TPDO data sent by the GCAN-IO-8000 coupler will be sent according to the following table. For example, when the GCAN-IO-8000 module node number is 3 and it is equipped with 3 GC-1008 modules and 1 GC-3804 module at the same time, you will receive two sets of data with frame IDs 0x183 and 0x283. The data correspondence is shown in the table below.

Frame ID	Frame data							
0x180+Node ID	number 1 GC-1008	number 2 GC-1008	number 3 GC-3804 first byte	number 3 GC-3804 second byte	number 3 GC-3804 third byte	number 3 GC-3804 fourth byte	number 3 GC-3804 fifth byte	number 3 GC-3804 sixth byte
0x280+Node ID	number 3 GC-3804 seventh byte	number 3 GC-3804 eighth byte	number 4 GC-1008	00	00	00	00	100

Table 4.4 Correspondence between multiple groups of GC-1008 modules and GC-3804 modules when mixed with CAN data

5. Technical specifications

Interface characteristics	
CANopen baud rate (bps)	1000k，500k，250k，125k，100k，50k，10k
PDO mode	Synchronization, looping, event-driven, polling
Number of bus terminal modules	32个
Fieldbus maximum number of bytes	96 byte input and 96 byte output
Digital I/O signal	384 input/output
Analog I/O signal	96 input/output
Configuration method	Automatic configuration
Number of PDO (CANopen)	4 Tx/4 Rx
Other CANopen features	Life protection/node protection, emergency objects, variable mapping, storage/reset
Bus interface	1 OPEN terminal interface, 4 pins
power supply	24V DC（-15%/+20%）
Input Current	70mA+ (total GC-bus current)/max 2.5A
Starting current	2.5 times continuous current
Recommended fuse capacity	≤10A
GC-bus supply current	500mA
Power supply	Max 24V DC/Max 10A
Electrical isolation	1500 Vrms
Environmental test	
Operating temperature	-40℃~+85℃
Working humidity	95%RH, no condensation
EMC test	EN 55024:2011-09 EN 55022:2011-12
Anti-vibration/impact resistance	EN 60068-2-6/EN 60068-2-27/29
Anti-electromagnetic interference/anti-electromagnetic radiation performance	EN 61000-6-2 /EN 61000-6-4
Protection level	IP 20
Basic Information	
Dimensions	100mm *69mm *44mm
weight	100g

6. GC series module selection table

GCAN-IO-8000 itself cannot perform complete control functions. A complete control system consists of a bus module controller (GCAN-IO-8000), several GC series terminal modules (GC-1008, GC-3804, etc.) and a terminal terminal module (GC-0001). Among them, GC series terminal modules need to be purchased separately from our company, and terminal terminal modules are included with GCAN-IO-8000.

GC series programmable controller expansion modules currently include: digital input expansion modules, digital output expansion modules, analog input expansion modules, and analog output expansion modules. The specific selection table is shown in the following table.

Species	Module	Characteristic	Signal	Channels
Digital input	GC-1008	Basic digital quantity	24V DC	8 channels
	GC-1502	Counter (100kHz max)	-	2channels
Digital output	GC-2008	Basic digital quantity	24V DC	8 channels
	GC-2204	Relay on	-	4 channels
	GC-2302	PWM（20Hz~20kHz）	-	2 channels
Analog input	GC-3604	Voltage input, 16 bits	-5V~+5V	4 channels
	GC-3624	Voltage input, 16 bits	-10V~+10V	4 channels
	GC-3644	Current input, 16 bits	0-20mA	4 channels
	GC-3654	Current input, 16 bits	4-20mA	4 channels
	GC-3664	Voltage input, 16 bits	0~+5V	4 channels
	GC-3674	Voltage input, 16 bits	0~+10V	4 channels
	GC-3804	2-wire PT100, 16-bit	Thermal resistance	4 channels
	GC-3822	3-wire PT100, 16-bit	Thermal resistance	2 channels

	GC-3844/3854/3864	K type / S type / T type thermocouple	Thermocouple	4 channels
Analog output	GC-4602	Voltage output, 16 bits	-5V~+5V	2 channels
	GC-4622	Voltage output, 16 bits	-10V~+10V	2 channels
	GC-4642	Current output, 16 bits	0-20mA	2 channels
	GC-4652	Current output, 16 bits	4-20mA	2 channels
	GC-4662	Voltage output, 16 bits	0~5V	2 channels
	GC-4672	Voltage output, 16 bits	0~10V	2 channels

Appendix A: CAN2.0A protocol frame format

CAN2.0A standard frame

The CAN standard frame information is 11 bytes, including two parts: information and data parts. The first 3 bytes are the information part.

	7	6	5	4	3	2	1	0
字节 1	FF	RTR	×	×	DLC（数据长度）			
字节 2	（报文识别码）				ID.10—ID.3			
字节 3	ID.2—ID.0			×	×	×	×	×
字节 4	数据 1							
字节 5	数据 2							
字节 6	数据 3							
字节 7	数据 4							
字节 8	数据 5							
字节 9	数据 6							
字节 10	数据 7							
字节 11	数据 8							

Byte 1 is frame information. The 7th bit (FF) indicates the frame format. In the standard frame, FF=0; the 6th bit (RTR) indicates the frame type. RTR=0 indicates the data frame, and RTR=1 indicates the remote frame; DLC indicates the data frame. The actual data length when framing.

Bytes 2 and 3 are message identification codes, 11 bits are valid.

Bytes 4 to 11 are the actual data of the data frame, which is invalid in the remote frame.

Appendix B: Introduction to CANopen Protocol

The CANopen protocol was developed in the late 1990s by the CiA organization (CAN-in-Automation) on the basis of CAL (CAN Application Layer). Once launched, it was widely recognized and applied in Europe. After several revisions to the CANopen protocol specification text, the stability, real-time, and anti-interference properties of the CANopen protocol have been further improved. And CiA has continuously introduced device sub-protocols in various industries, so that the CANopen protocol can be developed and promoted faster in various industries. At present, the CANopen protocol has been widely used in motion control, vehicle industry, motor drive, engineering machinery, marine shipping and other industries.

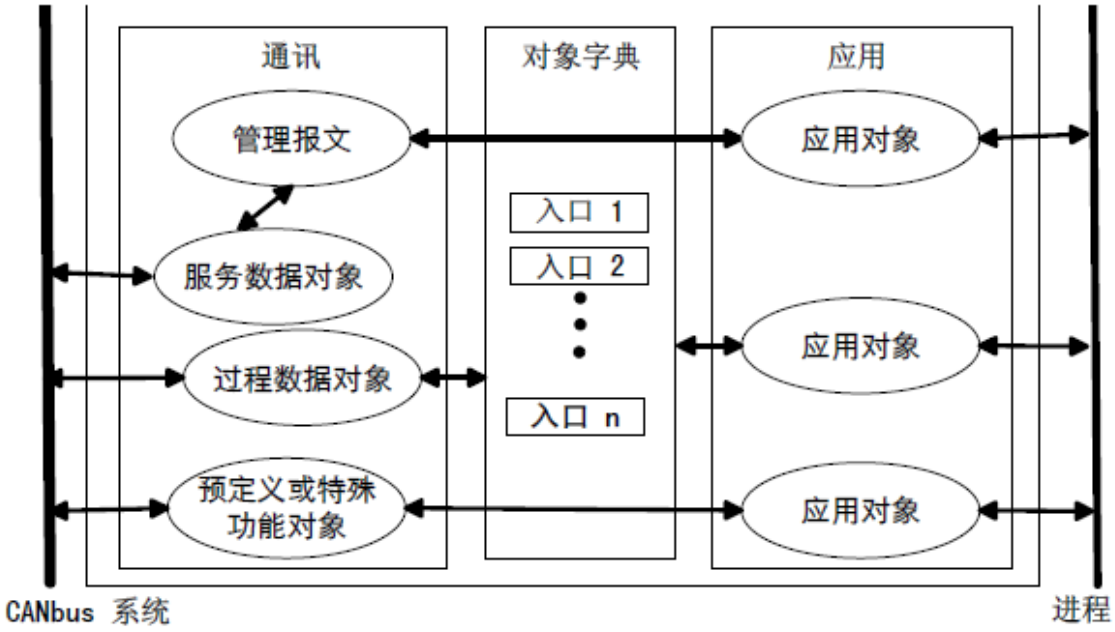


Figure A1 CANopen device structure

Figure A1 shows the CANopen device structure. The CANopen protocol is usually divided into three parts: user application layer, object dictionary, and communication.

B.1 Explanation of related terms and writing rules

1. Explanation of terms:
PDO：Process Data Object
TPDO：Transmit Process Data Object
RPDO：Receive Process Data Object
SDO：Service Data Object
NMT：Network Management
SYNC: Synchronization Objects
EMCY: Emergency Objects
OD：Object Dictionary

EDS：Electronic Data Sheet
CAN-ID: Controller Area Network-Identify
COB-ID: Communication Object-Identify
SSDO：Servers Service Data Object
DS：Draft Standard
2. Writing rules
In this manual, the writing of the object dictionary index and sub-index follow the rules shown in Figure A2 below, where the index is expressed in hexadecimal, the sub-index is expressed in decimal, and the index and sub-index are separated by spaces.

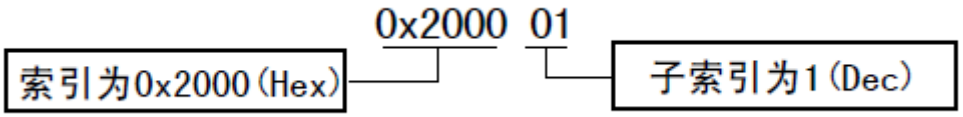


Figure A2 Index/sub-index writing rules

B.2 Predefined CAN identifier

Object	Function code	CAN-ID range
NMT network management commands	0000b	000h
Sync message	0001b	080h
Time Stamp message	0010b	100h
Emergency message	0001b	081h-0FFh
TPDO1 send process data object 1	0011b	181h-1FFh
RPDO1 Receive process data object 1	0100b	201h-27Fh
TPDO2 send process data object 2	0101b	281h-2FFh
RPDO2 Receive Process Data Object 2	0110b	301h-37Fh
TPDO3 send process data object 3	0111b	381h-3FFh
RPDO3 Receive Process Data Object 3	1000b	401h-47Fh
TPDO4 send process data object 4	1001b	481h-4FFh
RPDO4 receiving process data object 4	1010b	501h-57Fh
SDO Server-to-Client Service Data Object (Answer)	1011b	581h-5FFh
SDO Client-to-Server Service Data Object (Ask)	1100b	601h-67Fh

NMT error control Network management error control	1110b	701h-77Fh

B. 3 CANopen object dictionary

CANopen object dictionary (OD: Object Dictionary) is the core concept of CANopen protocol. The so-called object dictionary is an ordered group of objects. Each object is addressed with a 16-bit index value. This index value is usually called an index, and its effective range is between 0x1000 and 0x9FFF. To allow access to a single element in the data structure, an 8-bit index value is also defined. This index value is often referred to as a sub-index. Each CANopen device has an object dictionary. The object dictionary contains all the parameters describing the device and its network behavior. The object dictionary usually records these parameters in an electronic data file (EDS: Electronic Data Sheet), without the need to put these The parameters are recorded on paper. For the master node in the CANopen network, there is no need to access every object dictionary entry of the CANopen slave node.

The items in the CANopen object dictionary are described by a series of sub-protocols. The sub-protocol describes each object in the object dictionary its function, name, index, sub-index, data type, and whether this object is necessary, read and write attributes, etc., so as to ensure the compatibility of devices of the same type from different manufacturers. The core description sub-protocol of the CANopen protocol is DS301, which includes the application layer and communication structure description of the CANopen protocol. Other sub-protocols are supplements and extensions to the description text of the DS301 protocol. The CANopen protocol contains many sub-protocols, which are mainly divided into the following types.

1. Communication Profile

The communication sub-protocol describes the main form of the object dictionary and the communication objects and parameters in the object dictionary. This sub-protocol applies to all CANopen devices, and its index value ranges from 0x1000 to 0x1FFF.

2. Manufacturer-specific Profile

Manufacturer-defined sub-protocols. For special functions not defined in device sub-protocols, manufacturers can define object dictionary objects in this area according to requirements. Therefore, for different manufacturers, the definition of object dictionary items with the same index may not be the same, and the index value range is 0x2000~0x5FFF.

3. Device Profile

The device sub-protocol defines objects in the object dictionary for various types of devices. At present, there are more than ten kinds of sub-protocols defined for different types of devices, such as DS401, DS402, DS406, etc., and their index values range from 0x6000 to 0x9FFF.

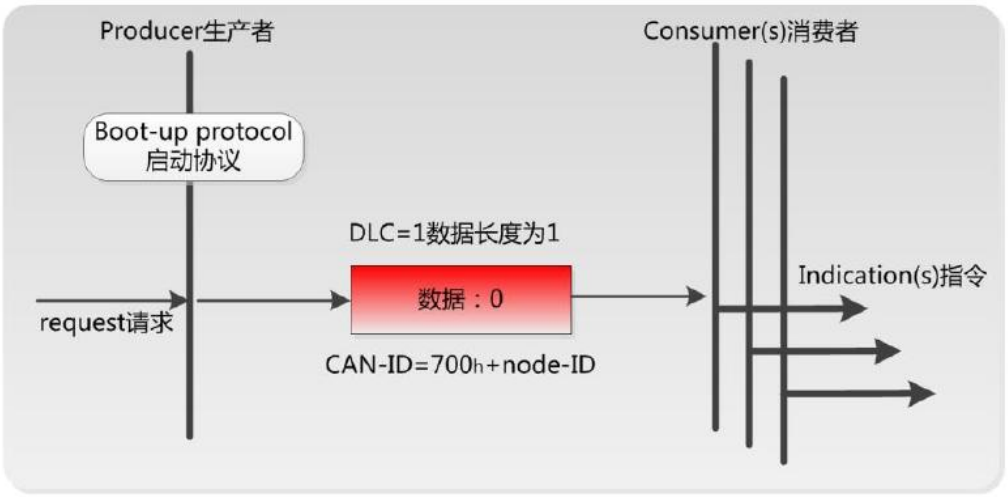
B.4 CANopen communication

The CANopen protocol mainly defines four objects: management message object

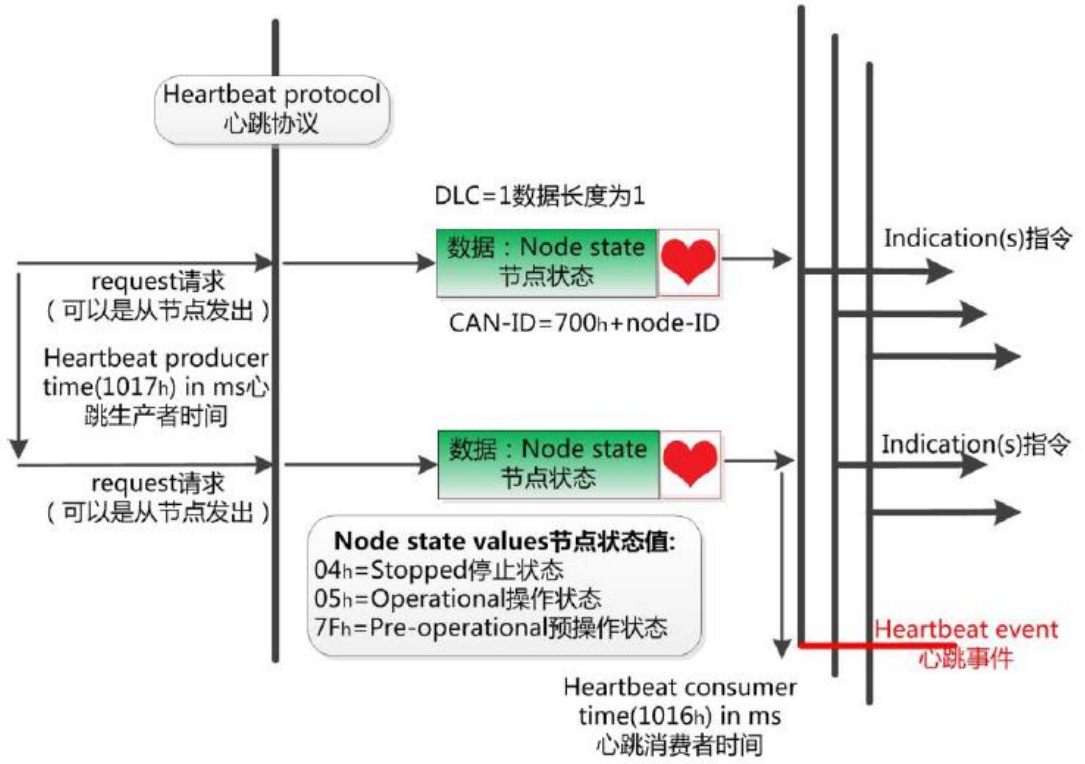
NMT (Network Management), service data object SDO (Service Data Object), process data object PDO (Process Data Object), predefined message or special function object.

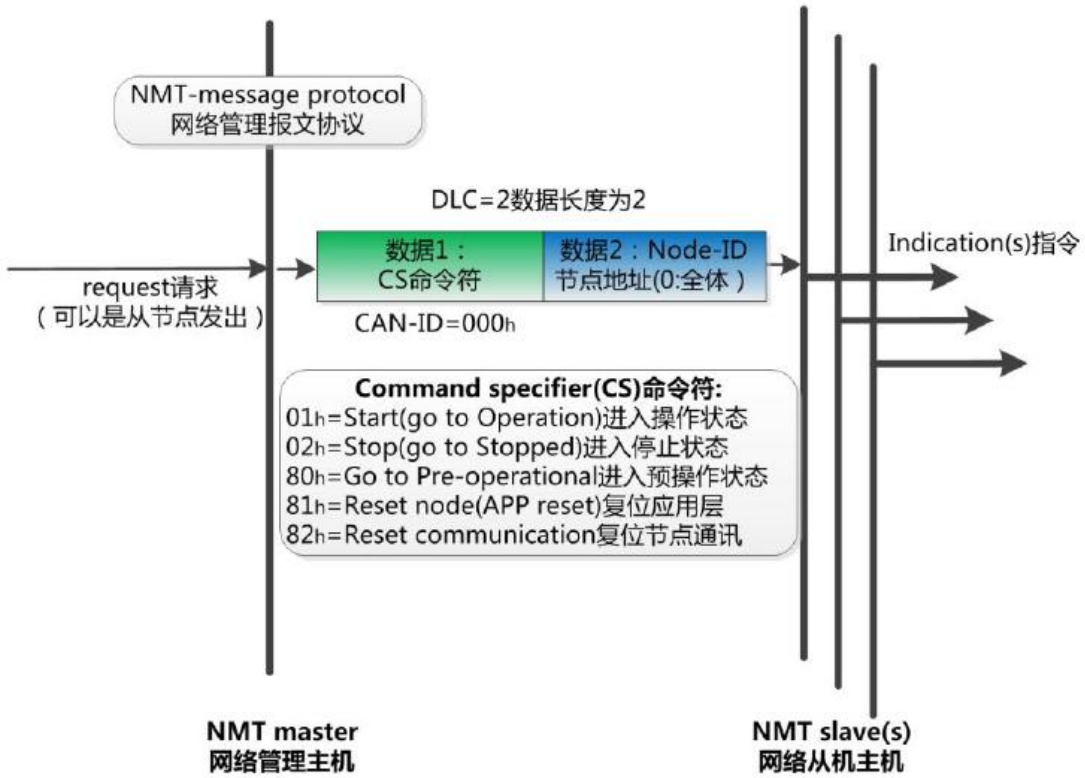
1. Network Management

Management messages are responsible for layer management, network management, and ID assignment services, such as initialization, configuration, and network management (including node protection). In network management, only one master node, one or more slave nodes are allowed in the same network, and follow the master-slave model. Through the NMT service, we can initialize, run, monitor, reset and stop the node. All nodes are considered as NMT slaves.



As shown in the figure above, for example, a CANopen slave device will send a data with a frame ID of 0x702 and a data of 0x00 after power on; it means that the device has been started and the node number is 2.

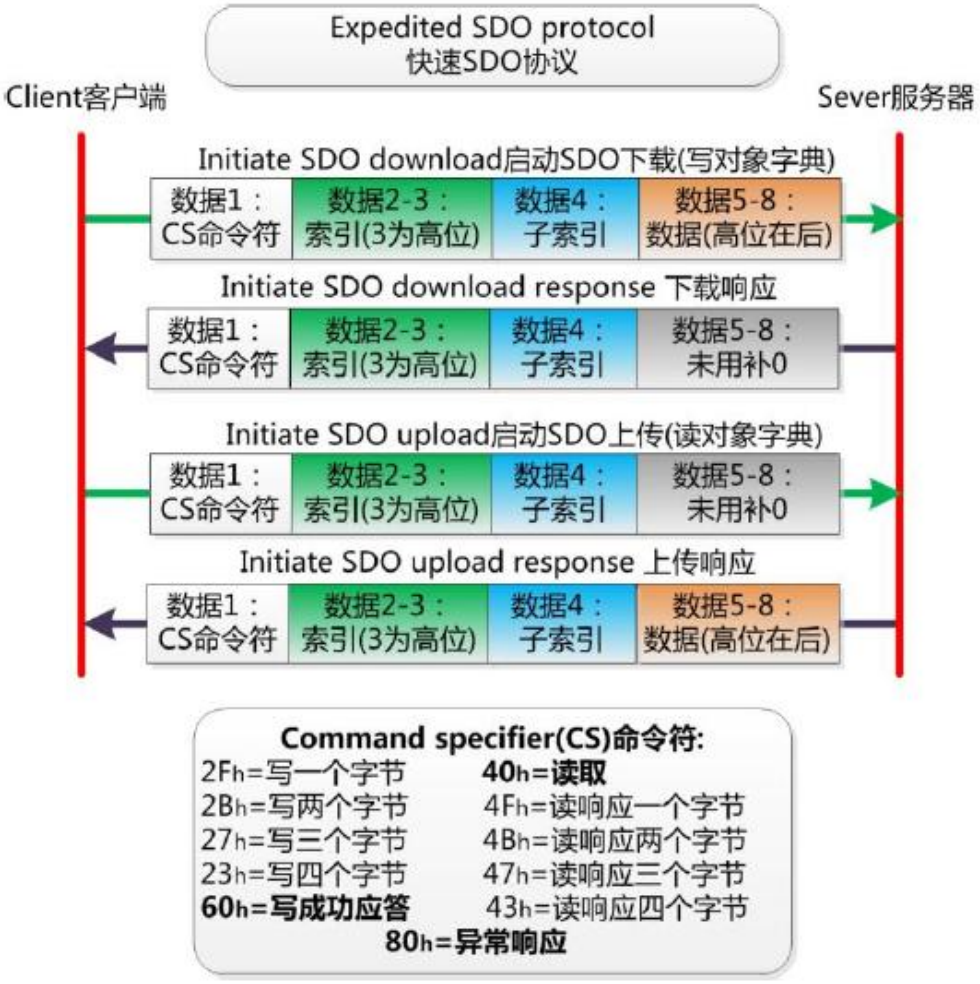




As shown in the figure above, for example, a CANopen master station sends a frame of data to the slave station, the frame ID is 0x000, the frame data is 0x01, 0x02, then this command can make the CANopen slave device with node number 2 enter the operating state .

2. Service Data Object

SDO is mainly used by the master node to configure the parameter of the slave node. Service confirmation is the biggest feature of SDO. It generates a response for each message to ensure the accuracy of data transmission. In a CANopen system, usually the CANopen slave node serves as the SDO server, and the CANopen master node serves as the client. The client can access the object dictionary on the data server through the index and sub-index. In this way, the CANopen master node can access the parameters of any object dictionary items of the slave node, and SDO can also transmit data of any length (when the data length exceeds 4 bytes, it is split into multiple messages for transmission).



3. Process Data Object

PDO is used to transmit real-time data. Its transmission model is the producer-consumer model, as shown in Figure A3. The data length is limited to 1~8 bytes. PDO communication objects have the following characteristics:

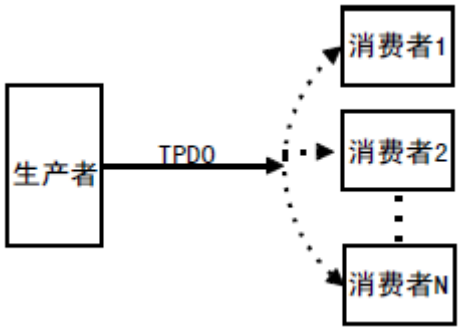
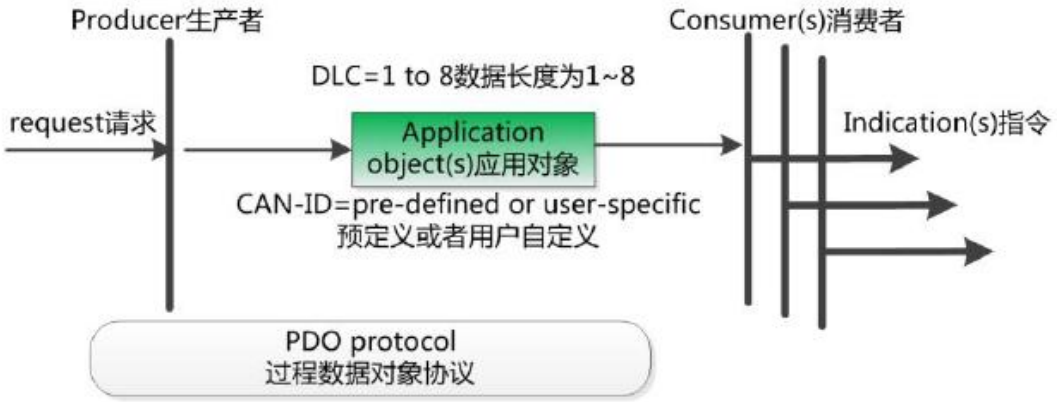


Figure A3 Producer consumer model

- PDO communication has no protocol, and the content of PDO data is defined by its CAN-ID (also called COB-ID);
- Each PDO is described by 2 objects in the object dictionary:
- PDO communication parameters, which define the COB-ID, transmission type and timing period used by the device;
- PDO mapping parameter, the mapping parameter contains a list of objects in the object dictionary, these objects are mapped to the corresponding PDO, including the length of the data (unit: bits), both producers and consumers must know this mapping parameter, Only then can the PDO content be correctly interpreted.
- The content of the PDO message is predefined. If the PDO supports variable PDO mapping, the PDO can be configured through SDO;
- PDO can have multiple transmission methods:
- Synchronous transmission (to achieve synchronization by receiving synchronization objects), synchronous transmission can be divided into aperiodic and periodic transmission. Acyclic transmission is pre-triggered by remote frames or pre-triggered by object-specific events specified in the device sub-protocol. Periodic transmission is achieved by receiving a synchronization object (SYNC), which can be set to trigger from 1 to 240 synchronization objects;
- Asynchronous transmission (triggered by a specific event), which can be triggered in two ways. The first is to trigger the transmission of PDO by sending a remote frame with the same COB-ID of PDO, and the second is specified in the device sub-protocol Object specific events to trigger (for example, timing transmission, data state change transmission, etc.).



4. Predefined messages or special function objects

Predefined messages or special function objects provide specific functions for CANopen devices, which is convenient for CANopen master to manage slave stations. In the CANopen protocol, the COB-ID has been predefined for special functions, which mainly have the following special messages:

- Synchronization (SYNC), the message object mainly realizes the synchronous transmission of the entire network, each node uses the synchronization message as the PDO synchronization trigger parameter, so the COB-ID of the synchronization message has a higher priority and the shortest Transmission time
- Time Stamp object (Time Stamp), providing a common time reference for each node;
- Emergency object (Emergency), when an error occurs in the device, the object is triggered, that is, the device internal error code is sent;
- Node/Life Guarding, the master node can obtain the status of the slave node through node protection. The slave node can obtain the status of the master node through life protection;
- Start the message object (Boot-up), send the object to the network after the initialization of the node, and enter the pre-operation state.

B.5 CANopen network configuration

In the CANopen protocol description text DS305, a network configuration protocol is defined, that is, the network configuration service LSS (Layer Setting Service), which uses the CANOpen module with the LSS master function to query or modify the CANOpen module with the LSS slave via the CAN bus. Certain parameters.

By using LSS, you can query or modify the following parameters:

- Node-ID of CANopen slave；
- Bit timing parameters of physical layer (baud rate)；
- LSS address (feature object 1018h).

Sales and Service

Shenyang Guangcheng Technology Co., Ltd.

Address: No. 135-21, Changqing South Street,
Hunnan District, Shenyang City, Liaoning Province
Postcode: 110000
Phone: +8613644001762
网址：www1.gcanbox.com
wechat:gckj777
whatsapp:+8613644001762
email:sygckj@gmail.com