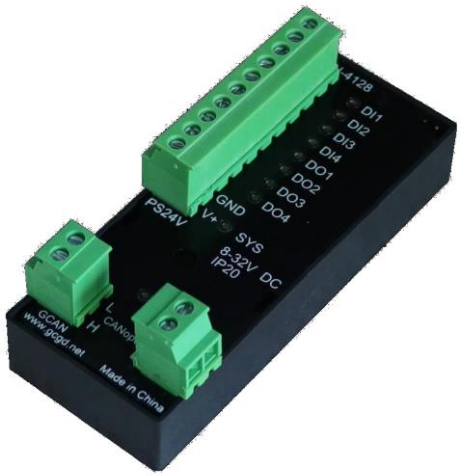
GCAN-4128

CAN/CANopen Remote IO - 4DI/4DO

User Manual



Document Version: V1.31 (2022/11/01)

Revision History

Version	Date	Reason
V1.00	2022/03/31	Document created
V1.10	2022/04/29	Adjusted document structure
V1.20	2022/05/26	Added timeout; object dictionary corrected
V1.30	2022/07/05	Document revision
V1.31	2022/11/01	Adjusted document structure

Contents

Revision History 2
Contents 3
1. Function Introduction 5
1.1 Overview 5
1.2 Features 5
1.3 Typical Applications 5
2. Device Installation 6
2.1 Device Dimensions 6
2.2 Interface Definition and Functions 6
3. Device Usage 7
3.1 Power Connection 7
3.4 System Status Indicators 7
4. Quick Start 9
5. Standard CANopen Mode 10
5.1 Node Status 10
5.2 NMT State Transition 10
5.3 PDO Commands 11
5.4 SDO Commands 12
5.4.1 SDO Write Object Dictionary 12
5.4.2 Change Operating Mode 12
5.4.3 Change Node ID 12
5.4.4 Change Baud Rate 13
5.4.5 Change the Display Position of Each DI 13
5.4.6 Change the Write Position of Each DO 13
5.5 Configuration Examples 13
5.6 Operation Examples 14
6. CAN Mode Parameter Configuration 15
6.1 Change DI Input Data Length and Cycle Time in CAN Mode 15
6.2 Change the Display Position of Each DI 15
6.3 Change DO Timeout and DO Data Length in CAN Mode 15
6.4 Change the Write Position of Each DO 15
6.5 Change the Input Data Frame ID 16
6.6 Change the Output Data Frame ID 16

6.7 Configuration Examples 16
7. CAN Mode Usage 17
7.1 Default Parameters 17
8. GCAN-4128 Object Dictionary 18
9. Technical Specifications 19
10. Disclaimer 20
1. Copyright and Translation Notice 20
2. Product Information and User Responsibility 20
3. Interpretation 20
Appendix A: CAN 2.0A Protocol Frame Format 21
CAN 2.0A Standard Frame 21
CAN 2.0B Extended Frame 21
Appendix B: Device Usage 22
Connection to CAN-bus 22
CAN-bus Terminating Resistor 23
Appendix C: Introduction to the CANopen Protocol 23
B.1 Terms and Writing Rules 24
B.2 Predefined CAN Identifiers 24
B.3 CANopen Object Dictionary 25
B.4 CANopen Communication 25
B.5 CANopen Network Configuration 29

1. Function Introduction

1.1 Overview

The GCAN-4128 module is an industrial-grade CAN/CANopen digital input/output module integrating one CAN-bus channel, two CAN interfaces, four digital input channels and four digital output channels. With the GCAN-4128 module, users can acquire digital input signals and output digital signals to control switches, realizing the connection between remote switching devices and the CAN network, and forming control nodes for digital signal acquisition and output in the CAN network.

The GCAN-4128 module is a key tool for digital signal acquisition and output. It is compact and easy to wire; it does not even need to be fixed inside a cabinet - as long as there is a small space, the cables can simply be routed in. The bus interface of the GCAN-4128 module integrates an isolation protection module, preventing damage to the module caused by instantaneous overvoltage or overcurrent.

1.2 Features

- CAN-bus supports CAN 2.0A and CAN 2.0B frame formats, compliant with ISO/DIS 11898.
- CAN-bus communication baud rate is freely programmable from 10 kbps to 1 Mbps.
- CAN-bus interface uses electrical isolation; isolation module insulation voltage: DC 1500 V.
- Powered by 8~32 V DC (standard 24 V recommended).
- Number of digital input channels: 4.
- Number of digital output channels: 4.
- Digital input high-level signal (logic 1): 5 V~32 V.
- Digital input low-level signal (logic 0): 0~2 V.
- Digital output signal, with output: supply voltage minus 0.5 V, rated output current 100 mA.
- Digital output signal, without output: 0~0.5 V, no output current.
- Operating temperature range: -40 °C ~ +85 °C.
- Operating humidity: ≤95% RH, non-condensing.
- Protection class: IP20.
- EMC test: EN50295.

1.3 Typical Applications

- Industrial field network data monitoring.
- Remote communication in coal mines and oil wells.
- CAN teaching and remote communication applications.
- CAN industrial automation control systems.
- Intelligent building control data broadcasting systems and other CAN-bus applications.

2. Device Installation

2.1 Device Dimensions

The outline dimensions of the device are (length) 78 mm x (width) 31.5 mm x (height, including terminal block) 26.5 mm, as shown in Figure 2.1.

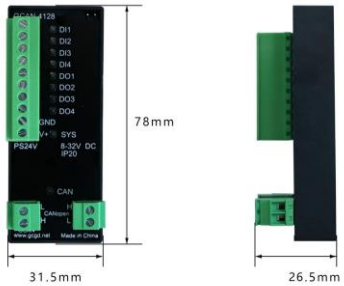


Figure 2.1 GCAN-4128 outline dimensions

2.2 Interface Definition and Functions

The GCAN-4128 module integrates four digital input interfaces, four digital output interfaces, one DC 8~32 V power interface, one CAN-bus channel, two CAN interfaces and one system reset interface. The positions of the digital, power and system reset interfaces are shown in Figure 2.2, and the interface definitions are given in Table 2.2.



Figure 2.2 Positions of 4-channel DI/DO, power and factory-reset interfaces

Table 2.2 4-channel DI/DO, power and system reset interface definitions

Pin (top to bottom)	Port	Name	Function
1-4	Digital	DI1~DI4	Digital input channels
5-8	Digital	DO1~DO4	Digital output channels
9	Power	GND	Negative terminal of 8~32 V DC supply
10	Power	V+	Positive terminal of 8~32 V DC supply
Top right	System reset	reset	Short when powered on to restore factory defaults

The CAN-bus channel of the GCAN-4128 module is led out through one 2-pin pluggable terminal block. The two CAN interfaces on the left and right can be used to connect one CAN-bus network or a CAN-bus device, and support the CANopen protocol. The CAN-bus interface positions are shown in Figure 2.3 and the definitions in Table 2.3.



Figure 2.3 CAN-bus interface positions

Table 2.3 CAN-bus interface definitions

Pin (left to right)	Port	Name	Function
CAN_L	CAN left	L	CAN_L signal line (CAN low)
CAN_H	CAN left	H	CAN_H signal line (CAN high)
CAN_H	CAN right	H	CAN_H signal line (CAN high)
CAN_L	CAN right	L	CAN_L signal line (CAN low)

3. Device Usage

3.1 Power Connection

The GCAN-4128 module supports the common +8~32 V DC industrial power supply. To ensure reliable operation, a regulated +12 V or +24 V DC power supply is recommended.

3.4 System Status Indicators

The GCAN-4128 module has one SYS indicator, eight IO indicators and one CAN indicator to show the operating status of the device. The specific functions and states of these three types of indicators are listed in Table 3.2.

LED	State	Indication	Figure
SYS	Fast green flash	Initialization passed, standby state	Figure 3.1
	Slow green flash	CANopen mode	
	Steady green	CAN mode	
	Off	Power supply failure	
CAN	Slow green flash	Small data volume, low rate	Figure 3.2
	Fast green flash	Large data volume, high rate	
	Red/green flashing	CAN interface data transmission error	
DI	Steady green	Corresponding DI has input	Figure 3.3
	Off	Corresponding DI has no input	
DO	Steady orange	Corresponding DO has output	Figure 3.4
	Off	Corresponding DO has no output	

Table 3.2 GCAN-4128 module LED status

- When the GCAN-4128 module is powered on, if the SYS indicator flashes green quickly and then slowly, the module is in the default CANopen mode; if the SYS indicator is steadily green, it is in CAN mode.
- When the GCAN-4128 module is powered on, the CAN indicator flashes green slowly for a small data volume and low rate, and very quickly for a large data volume and high rate.

- When a CAN interface data transmission error occurs, the CAN indicator flashes red and green; once the CAN interface data returns to normal, the CAN indicator flashes green.
- When a DI has an input, the corresponding DI LED is steadily green; with no input, it is off.
- When a DO has an output, the corresponding DO LED is steadily orange; with no output, it is off.

Figure 3.1 shows the GCAN-4128 SYS LED status; Figure 3.2 shows the CAN LED status.

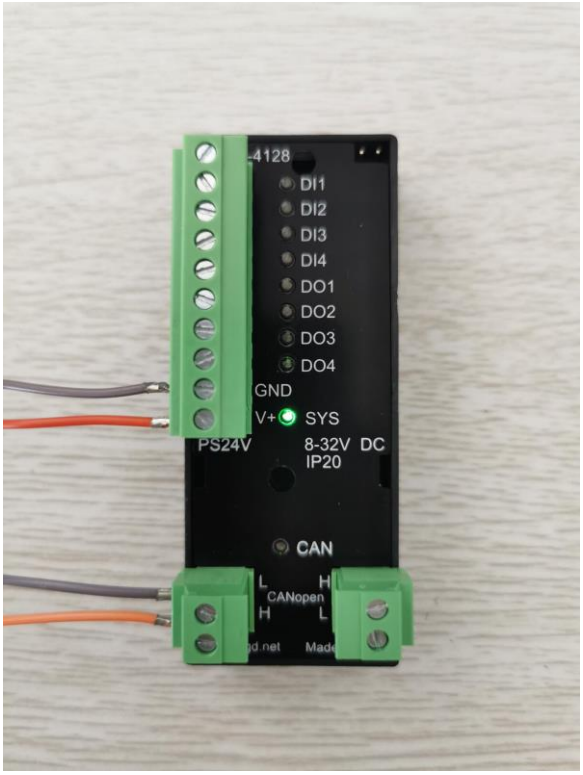


Figure 3.1 GCAN-4128 SYS LED status

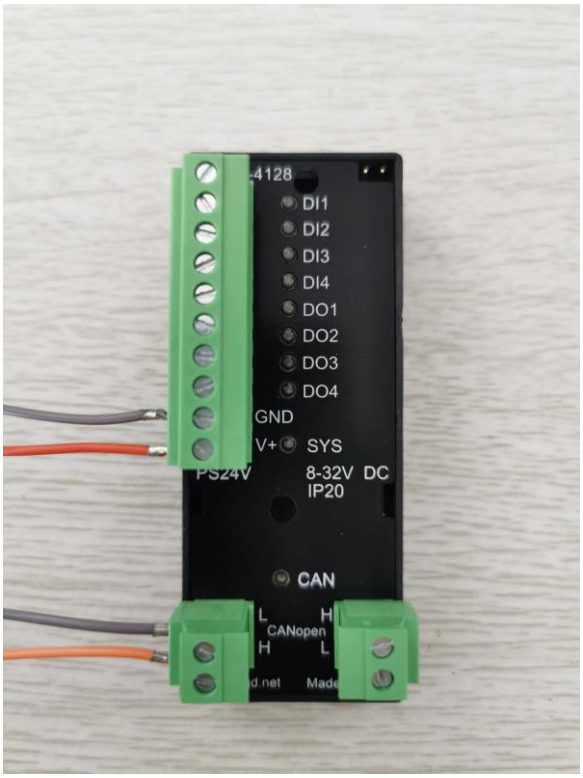


Figure 3.2 GCAN-4128 CAN LED status

As shown in Figure 3.3, DI1 of the GCAN-4128 module has an input while DI2-DI4 have no input; as shown in Figure 3.4, DO1 has an output while DO2-DO4 have no output.

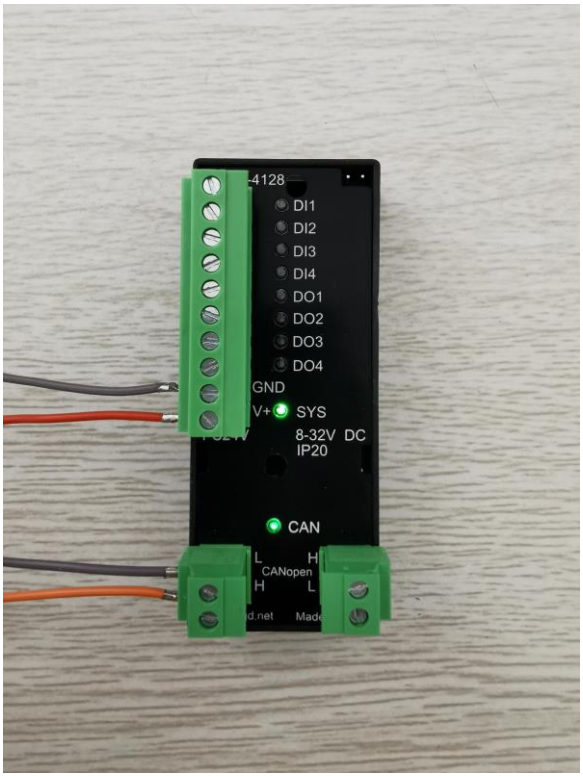


Figure 3.3 GCAN-4128 DI LED status



Figure 3.4 GCAN-4128 DO LED status

4. Quick Start

The GCAN-4128 module supports two modes, CANopen and CAN, with standard CANopen mode as default and a baud rate of 500 kbps. The instructions are as follows:

After power-on, the GCAN-4128 module automatically switches to state 05 (Operational) of CANopen mode. By default, immediately after power-on the module acquires digital signals and can control (write) the outputs.

For example, the default Node ID of the GCAN-4128 is 1 (all node IDs below are 1), so the master device can receive a node-status message with frame ID 0x701, standard frame, data 0x05.

In this state, TPDO1 indicates the digital DI status. Its frame ID is 0x180 + Node ID, and by default the first data byte is the digital input status, with DI4 in the high bit and DI1 in the low bit. RPDO1 indicates the control (write) status of the digital outputs; its frame ID is 0x200 + Node ID, and by default the digital output is controlled (written) by one byte, with DO4 in the high bit and DO1 in the low bit. See section 5.3 for details.

A typical CAN frame monitor capture in this state is shown below.

No.	Frame Interval (us)	Name	Frame ID	Frame Type	Frame Format	DLC	Data	Frame Count
00000000	000.020.013	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	63
00000001	000.000.000	Receive	701	DATA	STANDARD	1	05	1

Figure 4.1 CAN frame monitor - heartbeat and node status

5. Standard CANopen Mode

5.1 Node Status

The GCAN-4128 module complies with the standard CANopen CiA 301 protocol and is a standard CANopen slave device. After startup, the GCAN-4128 actively sends a data frame to the master with frame ID 0x700 + Node ID. For example, with the default Node ID of 1, the master can receive a node-status message with frame ID 0x701 and frame data 0x7F.

Object dictionary 0x100C, Guard Time, defaults to 0, so the timeout function is disabled by default; object dictionary 0x100D, Life Time Factor, defaults to 3.

After 0x100C and 0x100D are configured via SDO, the master actively sends a command to inquire about the slave node status (Node Guarding). The node-guarding command is a remote frame with ID 0x700 + Node ID. If slave 4128 does not receive the master's node-guarding message, after a delay of 0x100C (ms) x 0x100D the slave enters Pre-operational mode and stops all outputs.

The corresponding node-status capture is shown below.

No.	Frame Interval (us)	Name	Frame ID	Frame Type	Frame Format	DLC	Data	Frame Count
00000000	000.000.000	Receive	701	DATA	STANDARD	1	7F	1

Figure 5.1 Node-status frame capture

5.2 NMT State Transition

The GCAN-4128 module receives operation commands from the master. The command frame ID is 0x000 with DLC 2; the first data byte is the command specifier and the second is the node number (00 controls all nodes). For example, for Node ID 1, the command to enter the Operational state (01) is an NMT frame with ID 0x000 and data 0x01 0x01. When the master sends 01 01, the GCAN-4128 enters state 05 (Operational).

5.3 PDO Commands

The GCAN-4128 module uses PDO (Process Data Object) to acquire digital signals or write outputs. The specific position of each DI and DO channel can be freely configured (see 5.4).

TPDO1 indicates the digital DI status, with frame ID 0x180 + Node ID; by default the first data byte is the digital input status.

TPDO1 frame format:

COB-ID	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
0x180 + Node ID	DI status	NC	NC	NC	NC	NC	NC	NC

DI4 is in the high bit and DI1 in the low bit. For example, if the GCAN-4128 Node ID is set to 1, DI1 status is 1 and the other DIs are 0, the DI status data received by the CAN master is 0x01.

To control (write) the digital output status of the GCAN-4128, the master sends the RPDO1 command with frame ID 0x200 + Node ID, data length 1, and the first data byte holds the digital output status to be set.

RPDO1 indicates the change (write) of digital output status, with frame ID 0x200 + Node ID. By default the output is controlled by one byte, with DO4 in the high bit and DO1 in the low bit.

RPDO1 frame format:

COB-ID	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
0x200 + Node ID	DO status	NC	NC	NC	NC	NC	NC	NC

DO4 is in the high bit and DO1 in the low bit. For example, if Node ID is 1 and DO4 is to be set to 1 with all other DOs at 0, the master sends frame ID 0x201 with data 0x08. On the GCAN-4128, the orange LED of DO4 lights up, indicating an output at DO4.

The corresponding TPDO/RPDO frame captures are shown below.

No.	Frame Interval (us)	Name	Frame ID	Frame Type	Frame Format	DLC	Data	Frame Count
00000000	000.020.036	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	622
00000001	000.021.459	Send OK	701	DATA	STANDARD	1	Remote Request	60

Figure 5.2 TPDO1 frame capture

No.	Frame Interval (us)	Name	Frame ID	Frame Type	Frame Format	DLC	Data	Frame Count
00000000	007.000.031	Receive	701	DATA	STANDARD	1	05	364
00000001	001.000.000	Send OK	701	DATA	STANDARD	2	01 01	1

Figure 5.3 TPDO1 data example

No.	Frame Interval (us)	Name	Frame ID	Frame Type	Frame Format	DLC	Data	Frame Count
00000000	000.020.015	Receive	181	DATA	STANDARD	8	01 00 00 00 00 00 00 00	44048
00000001	001.000.063	Receive	701	DATA	STANDARD	1	05	880

Figure 5.4 RPDO1 frame capture

Save Data | Real-time Save | Pause Display | Display Mode | Clear | Filter Settings | Advanced Mask | Show Error Frames | Error Frames:0

No.	Frame Interval (us)	Name	Frame ID	Frame Type	Frame Format	DLC	Data	Frame Count
00000000	001.000.056	Receive	701	DATA	STANDARD	1	05	64
00000001	000.020.015	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	2396
00000002	000.000.000	Send OK	201	DATA	STANDARD	1	08	1

Figure 5.5 RPDO1 data example

5.4 SDO Commands

The Service Data Object (SDO) is mainly used to access the object dictionary of a node, allowing direct read/write configuration of the GCAN-4128 parameters. The configuration takes effect after power cycling.

In the frame tables below, Byte 0 is the command specifier (CS), Bytes 1-2 are the object dictionary index (low byte first, high byte second), Byte 3 is the subindex, and Bytes 4-7 are the data (low to high unless noted).

5.4.1 SDO Write Object Dictionary

SDO write command for a 32-bit data type:

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x23	08 / 02	24	00	low			high

SDO write command for a 16-bit data type:

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x2B	idx	24	00	low	high	0x00	0x00

SDO write command for an 8-bit data type:

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x2F	idx	24	00	0x00	0x00	0x00	value

Slave response - write successful:

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x60	idx	24	00	low	high		

Slave response - write failed:

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x80	idx	24	00	low	high		

5.4.2 Change Operating Mode

The GCAN-4128 can change its operating mode by writing a parameter via SDO. This parameter is at object dictionary index 0x2400, subindex 0x00, data type Unsigned8.

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x2F	00	24	00	0x00	0x00	0x00	mode

Data	Mode
0x01	CAN mode
0x02	Default mode
0x03	Standard CANopen mode

5.4.3 Change Node ID

The Node ID can be changed by writing a parameter via SDO. This parameter is at object dictionary index 0x2407, subindex 0x00, data type Unsigned8. The Node ID range is 1~127 (hexadecimal 0x01~0x7F).

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x2F	07	24	00	0x00	0x00	0x00	node ID

5.4.4 Change Baud Rate

The baud rate can be changed by writing a parameter via SDO. This parameter is at object dictionary index 0x2408, subindex 0x00, data type Unsigned32.

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x23	08	24	00	baud			

The relationship between data and baud rate is listed below; other values are invalid.

Data	Baud rate	Data	Baud rate
0x00	1000 kbps	0x05	100 kbps
0x01	800 kbps	0x06	50 kbps
0x02	500 kbps	0x07	20 kbps
0x03	250 kbps	0x08	10 kbps
0x04	125 kbps		

5.4.5 Change the Display Position of Each DI

The display position of each DI can be changed by writing a parameter via SDO. This parameter is at object dictionary index 0x2404, subindex 0x00, data type Unsigned32. The maximum display position is bit 64. In hexadecimal, the minimum value is 0x01 and the maximum is 0x40; 0x00 or values above 0x40 are invalid.

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x23	04	24	00	DI1	DI2	DI3	DI4

5.4.6 Change the Write Position of Each DO

The write position of each DO can be changed by writing a parameter via SDO. This parameter is at object dictionary index 0x2403, subindex 0x00, data type Unsigned32. The maximum position is bit 64. In hexadecimal, the minimum value is 0x01 and the maximum is 0x40; 0x00 or values above 0x40 are invalid.

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x23	03	24	00	DO1	DO2	DO3	DO4

5.5 Configuration Examples

CANopen mode configuration example (Node ID 1), after power-on in default mode:

1. Configure the module for standard CANopen mode. Message: ID 0x601, data 2F 00 24 00 03 00 00 00, sets the module to CAN mode.

2. Change the Node ID. Message: ID 0x601, data 2F 07 24 00 03 00 00 00, sets the Node ID to 3.

3. Change the baud rate. Message: ID 0x601, data 23 08 24 00 01 00 00 00, sets 800 kbps.

4. Change the display position of each DI. Message: ID 0x601, data 23 04 24 00 02 14 28 3C: DI4 at 0x3C (decimal bit 60), DI3 at 0x28 (bit 40), DI2 at 0x14 (bit 20), DI1 at 0x02 (bit 2).

5. Change the write position of each DO. Message: ID 0x601, data 23 03 24 00 02 14 28 3C: DO4 at 0x3C (bit 60), DO3 at 0x28 (bit 40), DO2 at 0x14 (bit 20), DO1 at 0x02 (bit 2).

After power cycling, the configuration takes effect. The standard CANopen mode can also be configured directly.

The following captures illustrate the CANopen configuration examples.

No.	Frame Interval (us)	Name	Frame ID	Frame Type	Frame Format	DLC	Data	Frame Count
00000377	000.020.012	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000378	000.019.968	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000379	000.020.011	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000380	000.020.003	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000381	000.019.993	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000382	000.020.011	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000383	000.019.993	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000384	000.020.017	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000385	000.020.015	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000386	027.817.845	Receive	701	DATA	STANDARD	1	05	1
00000387	027.816.677	Send OK	601	DATA	STANDARD	8	2F 00 24 00 03 00 00 00	1
00000388	009.740.982	Receive	581	DATA	STANDARD	8	60 00 24 00 02 00 00 00	1
00000389	026.087.995	Send OK	601	DATA	STANDARD	8	27 07 24 00 03 00 00 00	1
00000390	026.087.547	Receive	581	DATA	STANDARD	8	60 07 24 00 01 00 00 00	1
00000391	011.699.125	Send OK	601	DATA	STANDARD	8	23 08 24 00 01 00 00 00	1
00000392	011.698.345	Receive	581	DATA	STANDARD	8	60 08 24 00 02 00 00 00	1
00000393	013.783.920	Send OK	601	DATA	STANDARD	8	23 04 24 00 01 14 28 3C	1
00000394	013.782.845	Receive	581	DATA	STANDARD	8	60 04 24 00 02 02 03 04	1
00000395	006.878.284	Send OK	601	DATA	STANDARD	8	23 03 24 00 01 02 03 04	1
00000396	006.879.999	Receive	581	DATA	STANDARD	8	60 03 24 00 01 02 03 04	1

Figure 5.6 CANopen receive/configuration frame table

5.6 Operation Examples

CANopen mode operation example (Node ID 1), after re-powering following configuration:

1. NMT state transition. Message: ID 0x000, data 01 01, sets the module to state 05 (Operational). In this state, DI data can be read and DO data written.

2. DI display positions: DI4 at bit 60, DI3 at bit 40, DI2 at bit 20, DI1 at bit 2. TPDO1 has frame ID 0x180 + Node ID and is used to read DI data.

COB-ID	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
0x180+ Node ID	DI1 0x02	0x00	DI2 0x08	0x00	DI3 0x80	0x00	0x00	DI4 0x08

If channels 1 and 2 have inputs while channels 3 and 4 have no inputs, the data is as shown in the figure below.

The TPDO1 DI data example capture is shown below.

No.	Frame Interval (us)	Name	Frame ID	Frame Type	Frame Format	DLC	Data	Frame Count
00000000	000.020.018	Receive	181	DATA	STANDARD	8	02 00 08 00 00 00 00 00	133
00000001	001.000.031	Receive	701	DATA	STANDARD	1	05	2

Figure 5.7 TPDO1 DI data example

3. DO write positions: DO4 at bit 60, DO3 at bit 40, DO2 at bit 20, DO1 at bit 2. RPDO1 has frame ID 0x200 + Node ID and is used to write (set) DO data.

COB-ID	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
0x200+ Node ID	DO1 0x02	0x00	DO2 0x08	0x00	DO3 0x80	0x00	0x00	DO4 0x08

If channels 1 and 2 have no output while channels 3 and 4 have outputs, the data is as shown in the figure below.

The RPDO1 DO data example capture is shown below.

No.	Frame Interval (us)	Name	Frame ID	Frame Type	Frame Format	DLC	Data	Frame Count
00000000	000.020.017	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	324
00000001	001.000.038	Receive	701	DATA	STANDARD	1	05	7
00000002	000.000.000	Send OK	201	DATA	STANDARD	8	00 00 00 80 00 00 00 08	1

Figure 5.8 RPDO1 DO data example

6. CAN Mode Parameter Configuration

Note: In CAN mode there is only data transmission and reception; parameters cannot be configured. To reconfigure, short the reset button after power-on to enter CANopen mode.

6.1 Change DI Input Data Length and Cycle Time in CAN Mode

The DI data length and cycle time can be changed by writing a parameter via SDO. The data is at object dictionary index 0x2402, subindex 0x00, data type Unsigned32.

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x23	02	24	00	DI data length		cycle time	

6.2 Change the Display Position of Each DI

The display position of each DI can be changed by writing a parameter via SDO. The data is at object dictionary index 0x2404, subindex 0x00, data type Unsigned32. The maximum display position is bit 64; in hexadecimal the minimum is 0x01 and the maximum 0x40; 0x00 or values above 0x40 are invalid.

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x23	04	24	00	DI1	DI2	DI3	DI4

6.3 Change DO Timeout and DO Data Length in CAN Mode

The DO timeout and DO data length can be changed by writing a parameter via SDO. The data is at object dictionary index 0x2401, subindex 0x00, data type Unsigned32. By default, the module stops outputting 2 s after output begins.

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x2F	24	01	00	DO data length		timeout	

Timeout enable	Description
0x00	No timeout (default 0x07D0)
Timeout value	Timeout in ms, default 2000 ms

6.4 Change the Write Position of Each DO

The write position of each DO can be changed by writing a parameter via SDO. The data is at object dictionary index 0x2403, subindex 0x00, data type Unsigned32. The maximum position is bit 64; the minimum hexadecimal value is 0x01 and the maximum 0x40; 0x00 or values above 0x40 are invalid.

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x23	03	24	00	DO1	DO2	DO3	DO4

6.5 Change the Input Data Frame ID

The input frame ID can be changed by writing a parameter via SDO. The data is at object dictionary index 0x2406, subindex 0x00, data type Unsigned32. The input data frame ID refers to the frame ID of the input DI data in CAN mode.

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x23	06	24	00	input frame ID + frame format			

Frame format	Description
Highest bit of Unsigned32 = 1	Extended frame
Highest bit of Unsigned32 = 0	Standard frame

Examples: 23 06 24 00 01 00 00 00 sets the input frame ID to 0x001, standard frame. 23 06 24 00 00 08 00 80 sets the input frame ID to 0x00000800, extended frame.

6.6 Change the Output Data Frame ID

The output data frame ID can be changed by writing a parameter via SDO. The data is at object dictionary index 0x2405, subindex 0x00, data type Unsigned32. The output data frame ID refers to controlling the DO outputs by frame ID in CAN mode.

COB-ID	B0	B1	B2	B3	B4	B5	B6	B7
0x600+ Node ID	0x23	05	24	00	output frame ID + frame format			

Frame format	Description
Highest bit of Unsigned32 = 1	Extended frame
Highest bit of Unsigned32 = 0	Standard frame

Examples: 23 05 24 00 01 00 00 00 sets the output frame ID to 0x001, standard frame. 23 05 24 00 00 08 00 80 sets the output frame ID to 0x00000800, extended frame.

6.7 Configuration Examples

CAN mode configuration example (Node ID 1), after power-on in default mode:

1. Configure the module for CAN mode. Message: ID 0x601, data 2F 00 24 00 01 00 00 00.

2. Change DI input length and cycle time. Message: ID 0x601, data 23 02 24 00 08 E8 03 00: input length 8 bytes, cycle time 0x03E8 = 1000 ms.

3. Change the display position of each DI. Message: ID 0x601, data 23 04 24 00 02 14 28 3C: DI4 at bit 60, DI3 at bit 40, DI2 at bit 20, DI1 at bit 2.

4. Change DO timeout and DO data length. Message: ID 0x601, data 23 01 24 00 08 B8 0B 00: DO timeout 0x0BB8 = 3000 ms, DO data length 8 bytes.

5. Change the write position of each DO. Message: ID 0x601, data 23 03 24 00 02 14 28 3C: DO4 at bit 60, DO3 at bit 40, DO2 at bit 20, DO1 at bit 2.

6. Change the input data frame ID. Message: ID 0x601, data 23 06 24 00 11 00 00 00, input frame ID 0x11, standard frame.

7. Change the output data frame ID. Message: ID 0x601, data 23 05 24 00 33 00 00 00, output frame ID 0x33, standard frame.

7. CAN Mode Usage

7.1 Default Parameters

The GCAN-4128 is configured for CAN mode via 0x2400, with all other parameters at their default values (by default the first data byte holds the digital input/output status, DI4 / DO4 in the high bit and DI1 / DO1 in the low bit). After power cycling, data transmission and reception in CAN mode is as shown in the figure. Refer to chapter 6 for CAN mode parameters.

The input frame ID refers to the ID of the input DI, default 0x00000025; the output frame ID refers to the ID through which DO outputs can be controlled, default 0x00000017, with extended frames valid. The timeout defaults to 2 s, meaning the module stops outputting after 2 s; for continuous output, the send interval must be less than or equal to 2 s.

Note: In CAN mode there is only data transmission and reception; parameters cannot be configured. To reconfigure, short the reset button after power-on to enter CANopen mode.

The ECANTools capture of CAN mode data transmission and reception is shown below.

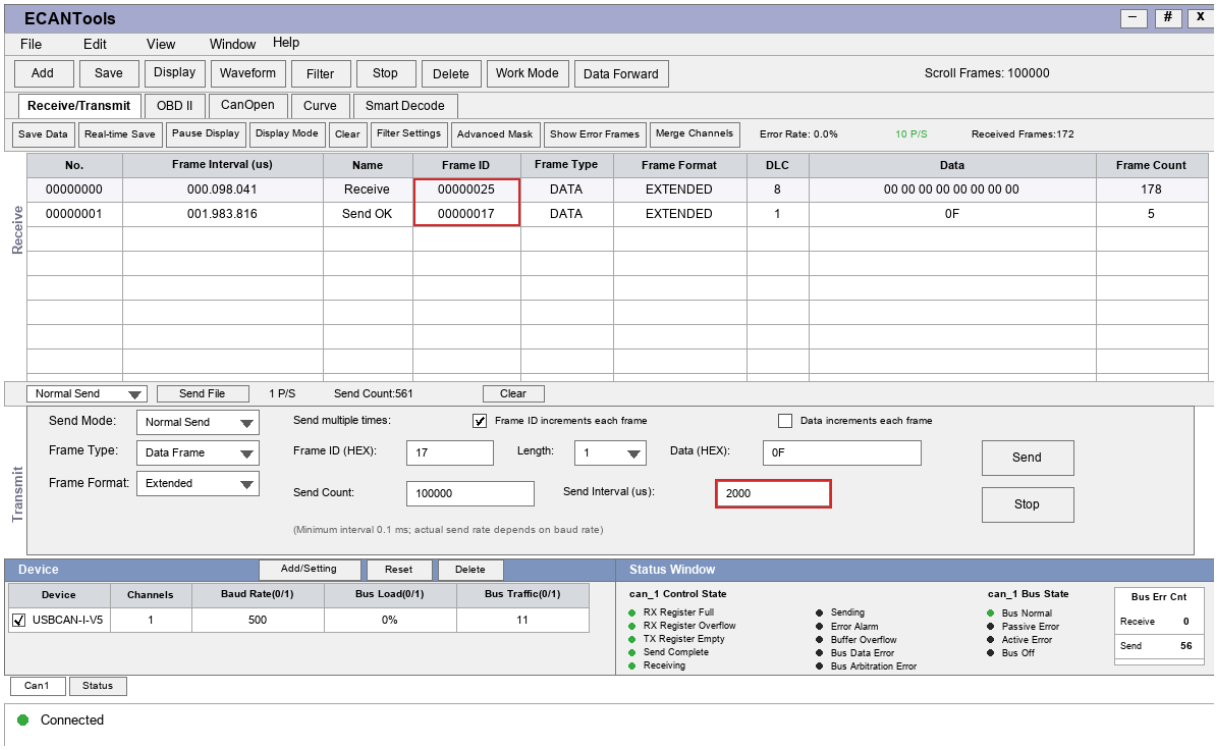


Figure 7.1 ECANTools software - CAN mode data communication

The CAN-mode frame configuration example is shown below.

No.	Frame Interval (us)	Name	Frame ID	Frame Type	Frame Format	DLC	Data	Frame Count
00000661	000.020.008	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000662	000.020.001	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000663	000.020.016	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000664	000.019.968	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000665	000.020.015	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000666	000.020.015	Receive	181	DATA	STANDARD	8	00 00 00 00 00 00 00 00	1
00000667	114.333.692	Send OK	601	DATA	STANDARD	8	2F 00 24 00 01 00 00 00	1
00000668	026.571.448	Receive	581	DATA	STANDARD	8	60 00 24 00 02 00 00 00	1
00000669	023.113.530	Send OK	601	DATA	STANDARD	8	23 02 24 00 08 E8 03 00	1
00000670	023.113.381	Receive	581	DATA	STANDARD	8	60 02 24 00 02 63 00 00	1
00000671	043.562.740	Send OK	601	DATA	STANDARD	8	23 03 24 00 01 14 28 3C	1
00000672	000.000.000	Receive	581	DATA	STANDARD	8	60 00 24 00 02 03 04 00	1
00000673	014.896.066	Send OK	601	DATA	STANDARD	8	23 01 24 00 08 B8 08 00	1
00000674	014.000.000	Receive	581	DATA	STANDARD	8	60 01 24 00 08 D8 07 00	1
00000675	014.561.411	Send OK	601	DATA	STANDARD	8	23 03 24 00 02 14 28 3C	1
00000676	000.000.000	Receive	581	DATA	STANDARD	8	60 03 24 00 02 14 28 3C	1
00000677	000.000.000	Receive	06004007	DATA	EXTENDED	8	00 00 00 00 00 00 00 00	1
00000678	024.290.312	Send OK	601	DATA	STANDARD	8	23 06 24 00 11 00 00 00	1
00000679	000.000.000	Receive	581	DATA	STANDARD	8	60 06 24 00 25 00 00 00	1
00000680	005.873.255	Send OK	601	DATA	STANDARD	8	23 05 24 00 33 00 00 00	1
00000681	005.873.545	Receive	581	DATA	STANDARD	8	60 05 24 00 17 00 00 80	1

Figure 7.2 CAN frame configuration example

8. GCAN-4128 Object Dictionary

Communication parameter area:

Index	Sub	Name	Type	Attr.	Default	Description
0x1000	-	Device Type	UINT32	RO	0x000A0011	Device type
0x1001	-	Error Register	UINT8	RO	0	Current error type
0x1003	0	number of errors	UINT8	RO	0	-
	1~4	standard error field	UINT32	RO	0	History of emergency error codes
0x1005	-	COB-ID SYNC	UINT32	RW	0x80	-
0x1006	-	Communication Cycle Period	UINT16	RW	0x2710	Communication cycle period
0x1007	-	Sync Window Length	UINT32	RW	0	-
0x1008	-	GCAN-4128 Name	STRING	CONST	GCAN-4128	GCAN-4128
0x1009	-	hardware version	STRING	CONST	V1.01	Hardware version
0x100A	-	software version	STRING	CONST	5.18	Software version
0x100C	-	Guard Time	UINT16	RW	0x00	Node guarding time
0x100D	-	Life Time Factor	UINT8	RW	0x03	-
0x1016	0	Number Of Entries	UINT8	RO	0x1	-
	1	Consumer Heartbeat Time #1	UINT32	RW	0x0	-
0x1017	-	Producer Heartbeat Time	UINT16	RW	0x03E8	-
0x1018	0	Highest sub-index supported	UINT8	RO	4	-
	1	Vendor-ID	UINT32	RO	0x00000449	Vendor code registered with CiA
	2	Product code	UINT32	RO	0x4128	Product code
	3	Revision number	UINT32	RO	0x00000000	Revision number
	4	Serial number	UINT32	RO	0x00000001	Serial number

RPDO communication parameters:

Index	Sub	Name	Type	Attr.	Default	Description
0x1400	0	Number of entries	UINT8	RO	3	-
	1	COB-ID	UINT32	RW	NodeID + 0x200	COB-ID used by RPDO
	2	Transmission type	UINT8	RW	255	Transmission type
	3	Inhibit timer	UINT16	RW	0	-
	5	Event timer	UINT16	RW	0x0	-
0x1600	0	Number of mapped objects	UINT8	RO	0x1	-
	1	1. mapped object	UINT32	RO	0x20100508	-

TPDO communication parameters:

Index	Sub	Name	Type	Attr.	Default	Description
0x1800	0	largest subindex supported	UINT8	RO	0x5	-

	1	COB-ID used	UINT32	RW	NodeID + 0x180	COB-ID used by TPDO
	2	transmission type	UINT8	RW	0xFF (255)	Transmission type
	3	inhibit time	UINT16	RW	0x64	-
	5	event timer	UINT16	RW	0x01	-
0x1A00	0	number of mapped objects	UINT8	RO	0x4	-
	1	PDO mapping 1 app. object	UINT32	RO	0x20100110	-
	2	PDO mapping 2 app. object	UINT32	RO	0x20100210	-
	3	PDO mapping 3 app. object	UINT32	RO	0x20100310	-
	4	PDO mapping 4 app. object	UINT32	RO	0x20100410	-

Device status:

Index	Sub	Name	Type	Attr.	Default	Description
0x2400	00	Type	UINT8	RW	0x0002	CAN/CANopen mode switch
0x2401	00	Canrec Timeout Len	UINT32	RW	0x0007D008	Timeout, output data length
0x2402	00	Cansend Time Len	UINT32	RW	0x0006308	Input data length, cycle time
0x2403	00	Output Map Bit	UINT32	RW	0x04030201	Output position
0x2404	00	Input Map Bit	UINT32	RW	0x04030201	Input position
0x2405	00	Canrec Id	UINT32	RW	0x80000017	Output CAN ID
0x2406	00	Casend Id	UINT32	RW	0x80000025	Input CAN ID
0x2407	00	Nodeid	UINT8	RW	0x01	Node ID
0x2408	00	Canbaud	UINT32	RW	0x00002	CAN baud rate

9. Technical Specifications

Category	Item	Specification
Connection	DI/DO interface	Terminal block
	CAN interface	Terminal block
Interface features	CAN interface	Compliant with ISO 11898, supports standard CANopen protocol and CAN 2.0A/B
	CAN baud rate	1000K, 800K, 500K, 250K, 125K, 100K, 50K, 20K, 10K
	Electrical isolation	1500 V, DC-DC
	CAN terminating resistor	Not integrated; add between CAN_H and CAN_L if required
Power supply	Supply voltage	+8~32 V DC
	Supply current	Max. 140 mA (quiescent current: 40 mA)
Environmental	Operating temperature	-40 °C ~ +85 °C
	Operating humidity	15%~90% RH, non-condensing
	EMC test	EN 50295
	Protection class	IP 20
General	Dimensions	78 mm x 31.5 mm x 26.5 mm
	Weight	100 g

10. Disclaimer

Thank you for purchasing GCAN series hardware and software products.

1. Copyright and Translation Notice

The original Chinese-language version of this document was prepared by Shenyang Guangcheng Technology Co., Ltd., which retains the copyright to the original document and its contents.

This English version has been independently translated and prepared by Shenzhen MoCAN Electric Co., Ltd. based on its understanding of the original Chinese documentation. It is intended solely for overseas customer reference and technical support purposes. It has not been officially reviewed, verified or approved by Shenyang Guangcheng Technology Co., Ltd., and should not be regarded as an official or authoritative English version of the original documentation.

The English translation, editorial work and original layout elements created by Shenzhen MoCAN Electric Co., Ltd. are protected by applicable copyright laws. No part of this English version may be reproduced, redistributed, republished or used for commercial purposes without prior written permission from Shenzhen MoCAN Electric Co., Ltd., except as permitted by applicable law.

GCAN is a registered trademark of Shenyang Guangcheng Technology Co., Ltd.

2. Product Information and User Responsibility

Product specifications, technical parameters, functions and other information provided in this document are for reference purposes and may be subject to change without prior notice.

Users are responsible for evaluating the suitability of the products for their specific applications and ensuring proper installation, configuration and operation in accordance with the relevant technical documentation and applicable safety requirements.

For product selection, technical clarification or application-specific requirements, users are advised to consult the manufacturer or an authorized supplier before making purchasing or implementation decisions.

3. Interpretation

This English document is provided as a reference translation of the original Chinese-language documentation. While reasonable efforts have been made to ensure the accuracy and consistency of the translation, differences in technical terminology, interpretation or expression may occur.

In the event of any discrepancy, ambiguity or inconsistency between this English version and the original Chinese-language documentation, the original Chinese-language version shall prevail.

For technical questions or clarification regarding the product specifications and intended functions, users are encouraged to contact Shenyang Guangcheng Technology Co., Ltd. or their authorized supplier.

Appendix A: CAN 2.0A Protocol Frame Format

CAN 2.0A Standard Frame

A CAN standard frame carries 11 bytes of information, consisting of two parts: the information part and the data part. The first three bytes form the information part.

Byte 1 is the frame information. Bit 7 (FF) indicates the frame format; in a standard frame FF = 0. Bit 6 (RTR) indicates the frame type: RTR = 0 for a data frame, RTR = 1 for a remote frame. DLC indicates the actual data length in a data frame.

Bytes 2 and 3 are the message identifier, of which 11 bits are valid.

Bytes 4 to 11 contain the actual data of a data frame and are invalid in a remote frame.

	7	6	5	4	3	2	1	0
Byte 1	FF	RTR	×	×	DLC (Data Length)			
Byte 2	(Message ID) ID.10-ID.3							
Byte 3	ID.2-ID.0			×	×	×	×	×
Byte 4	Data 1							
Byte 5	Data 2							
Byte 6	Data 3							
Byte 7	Data 4							
Byte 8	Data 5							
Byte 9	Data 6							
Byte 10	Data 7							
Byte 11	Data 8							

Table A.1 CAN 2.0A standard frame format

CAN 2.0B Extended Frame

A CAN extended frame carries 13 bytes of information, consisting of the information part and the data part. The first five bytes form the information part.

Byte 1 is the frame information. Bit 7 (FF) indicates the frame format; in an extended frame FF = 1. Bit 4 (RTR) indicates the frame type: RTR = 0 for a data frame, RTR = 1 for a remote frame. DLC indicates the actual data length in a data frame.

Bytes 2 to 5 are the message identifier, of which the high 29 bits are valid.

Bytes 6 to 13 contain the actual data of a data frame and are invalid in a remote frame.

	7	6	5	4	3	2	1	0
Byte 1	FF	×	×	RTR	DLC (Data Length)			
Byte 2	(Message ID) ID.28-ID.21							
Byte 3	ID.20-ID.13							
Byte 4	ID.12-ID.5							
Byte 5	ID.4-ID.0					×	×	×
Byte 6	Data 1							
Byte 7	Data 2							
Byte 8	Data 3							
Byte 9	Data 4							
Byte 10	Data 5							
Byte 11	Data 6							
Byte 12	Data 7							
Byte 13	Data 8							

Table A.2 CAN 2.0B extended frame format

Appendix B: Device Usage

Connection to CAN-bus

To connect the GCAN-4128 module to the CAN bus, connect CAN_H to CAN_H and CAN_L to CAN_L to establish communication.

The CAN-bus network uses a linear topology. A 120-ohm terminating resistor must be installed at the two most distant ends of the bus; if there are more than two nodes, middle nodes do not need a 120-ohm terminating resistor. Branch connections should not exceed 3 m. The CAN-bus connection is shown in Figure B.1.

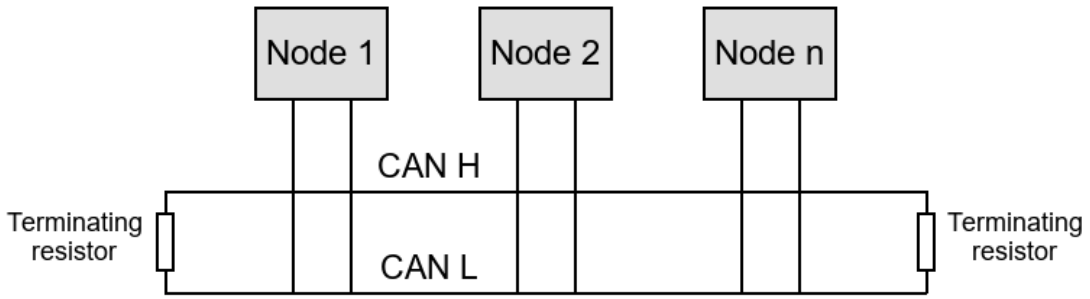


Figure B.1 CAN-bus network topology

Note: CAN-bus cable can be ordinary twisted pair or shielded twisted pair. The theoretical maximum communication distance depends mainly on the bus baud rate; see Table B.1. If the

communication distance exceeds 1 km, ensure the wire cross-section is greater than 1.0 mm2; the specific specification depends on the distance and is generally increased appropriately as the distance grows.

Table B.1 Baud rate vs. maximum bus length

Baud rate	Bus length	Baud rate	Bus length
1 Mbit/s	25 m	125 kbit/s	500 m
800 kbit/s	66 m	50 kbit/s	1.0 km
500 kbit/s	100 m	20 kbit/s	2.5 km
250 kbit/s	250 m	10 kbit/s	5.0 km

CAN-bus Terminating Resistor

To improve CAN communication reliability and eliminate signal reflection at the bus ends, a terminating resistor is usually added at the two most distant ends of the CAN-bus network, as shown in Figure B.2. The value of the terminating resistor is determined by the characteristic impedance of the transmission cable. For example, if the characteristic impedance of the twisted pair is 120 ohm, a 120-ohm terminating resistor should be integrated at both ends of the bus. If other nodes on the network use different transceivers, the terminating resistor must be calculated separately.

Figure B.2 shows the GCAN-4128 connected to another CAN node.

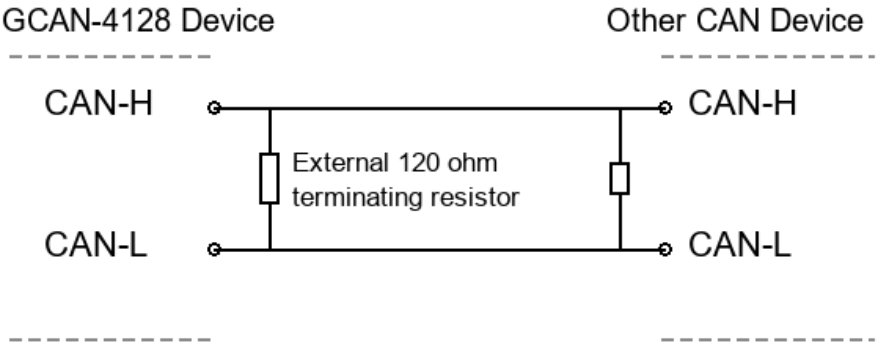


Figure B.2 GCAN-4128 connected to another CAN node

Note: The GCAN-4128 module does not integrate a 120-ohm terminating resistor. If there are more than two nodes, middle nodes do not need a 120-ohm terminating resistor. When one is needed, connect the two ends of the resistor to CAN_H and CAN_L respectively, as shown in Figure B.2.

Appendix C: Introduction to the CANopen Protocol

The CANopen protocol was developed in the late 1990s by the CiA organization (CAN-in-Automation) based on CAL (CAN Application Layer), and was widely recognized and applied in Europe soon after its release. Through several revisions of the CANopen specification, its stability, real-time performance and anti-interference capability have been further improved. CiA continues to introduce device sub-protocols for various industries, promoting faster development and adoption of CANopen. Today CANopen is widely used in motion control, vehicle industry, motor drives, construction machinery, shipbuilding and marine transport.

Figure B.3 shows the CANopen device structure. The CANopen protocol is generally divided into three parts: the user application layer, the object dictionary and communication.

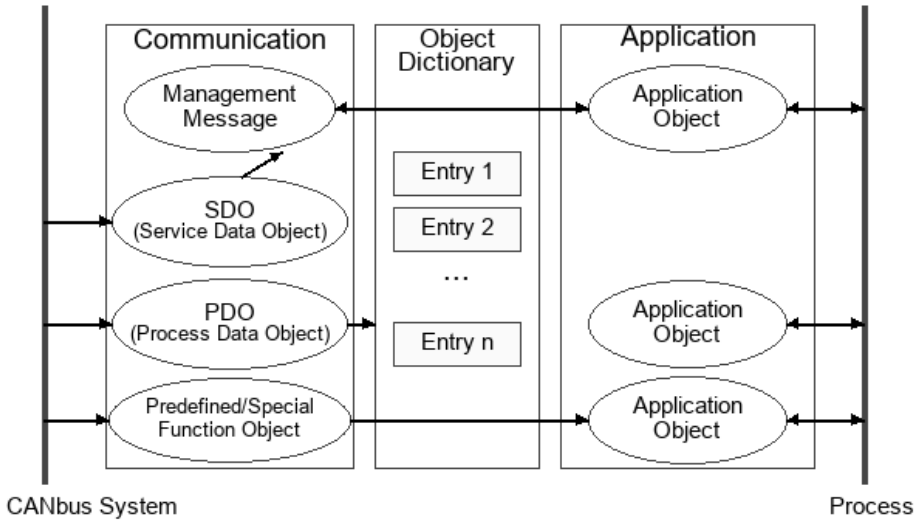


Figure B.3 CANopen device structure

B.1 Terms and Writing Rules

1. 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: Server Service Data Object.
- DS: Draft Standard.

2. Writing rules: In this manual, the object dictionary index and subindex are written according to the rule shown in Figure B.4, where the index is in hexadecimal, the subindex is in decimal, and the two are separated by a space.

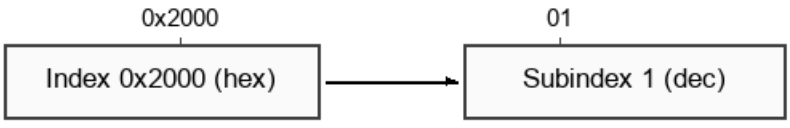


Figure B.4 Index/subindex writing rule

B.2 Predefined CAN Identifiers

Object	Function code	CAN-ID range
NMT network management command	0000b	000h
Sync message	0001b	080h
Time Stamp message	0010b	100h
Emergency message	0001b	081h-0FFh
TPDO1	0011b	181h-1FFh

RPDO1	0100b	201h-27Fh
TPDO2	0101b	281h-2FFh
RPDO2	0110b	301h-37Fh
TPDO3	0111b	381h-3FFh
RPDO3	1000b	401h-47Fh
TPDO4	1001b	481h-4FFh
RPDO4	1010b	501h-57Fh
SDO Server-to-Client (answer)	1011b	581h-5FFh
SDO Client-to-Server (request)	1100b	601h-67Fh
NMT error control	1110b	701h-77Fh

B.3 CANopen Object Dictionary

The CANopen Object Dictionary (OD) is the most core concept of the CANopen protocol. It is an ordered group of objects, each addressed by a 16-bit index value, usually called the index, with a valid range from 0x1000 to 0x9FFF. To allow access to individual elements in a data structure, an 8-bit index value is also defined, usually called the subindex. Every CANopen device has an object dictionary containing all parameters describing the device and its network behavior. These parameters are usually recorded in an Electronic Data Sheet (EDS) rather than on paper. For the master node in a CANopen network, it is not necessary to access every object dictionary entry of a slave node.

The entries in the CANopen object dictionary are described by a series of sub-protocols. Each sub-protocol describes the function, name, index, subindex, data type, whether the object is mandatory, and access attributes for every object, ensuring compatibility of similar devices from different manufacturers. The core descriptive sub-protocol is DS301, which covers the CANopen application layer and communication structure; other sub-protocols supplement and extend DS301. The main types are:

- Communication Profile: describes the main form of the object dictionary and its communication objects and parameters. It applies to all CANopen devices, index range 0x1000~0x1FFF.
- Manufacturer-specific Profile: for special functions not defined in device profiles, the manufacturer may define object dictionary objects in this area as needed. Definitions for the same index may differ between manufacturers, index range 0x2000~0x5FFF.
- Device Profile: defines object dictionary objects for different types of devices. More than ten profiles exist, e.g. DS401, DS402, DS406, index range 0x6000~0x9FFF.

B.4 CANopen Communication

The CANopen protocol mainly defines four types of objects: Network Management (NMT), Service Data Object (SDO), Process Data Object (PDO), and predefined or special-function objects.

1. Network Management (NMT): The management messages handle layer management, network management and ID allocation services, such as initialization, configuration and network management (including node guarding). In network management, only one master node and one or more slave nodes are allowed in a network, following a master-slave model. Through NMT services, nodes can be initialized, started, monitored, reset and stopped. All nodes are considered NMT slaves.

Figure B.5 illustrates the boot-up protocol: after power-on, a CANopen slave (the producer) sends a boot-up message with COB-ID 0x700 + node ID and data 0x00; consumers on the network receive this indication to learn that the node has started.

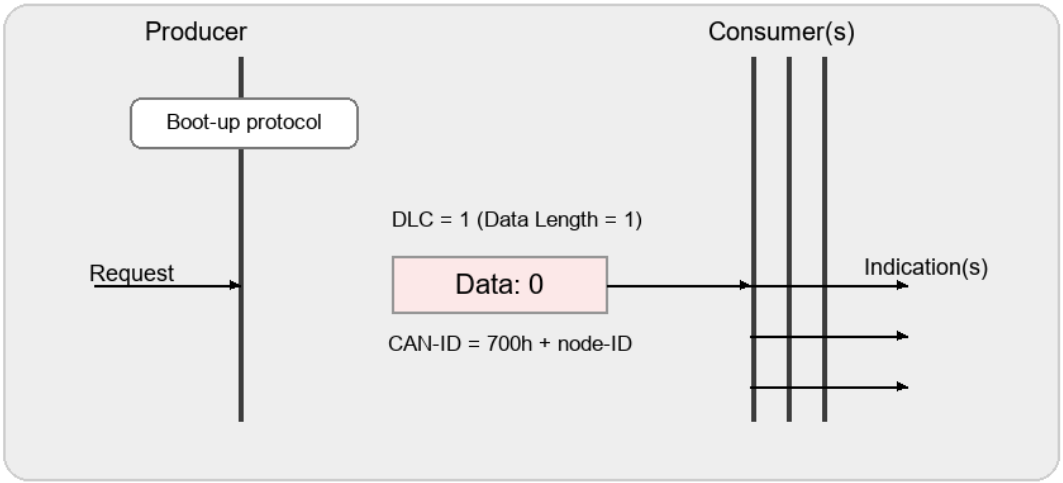


Figure B.5 Boot-up protocol

Figure B.6 shows the heartbeat mechanism and NMT node-state messages. A heartbeat producer periodically sends a CAN frame (COB-ID = 0x700 + node ID) to one or more consumers, which monitor the producer's state. The data byte indicates the node state (0x00 = boot-up, 0x04 = stopped, 0x05 = operational, 0x7F = pre-operational). The NMT master sends a 2-byte command (command specifier + node ID) with COB-ID 0x000; command specifiers include 0x01 (start remote node), 0x02 (stop remote node), 0x80 (pre-operational), 0x81 (reset node) and 0x82 (reset communication).

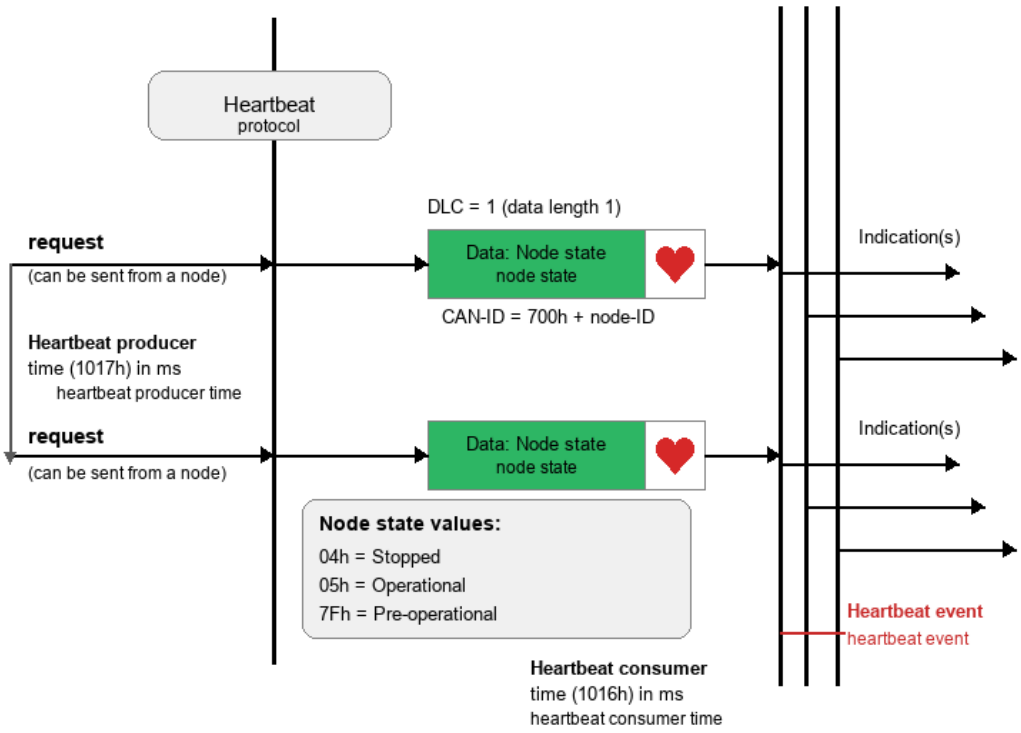


Figure B.6 Heartbeat protocol and node states

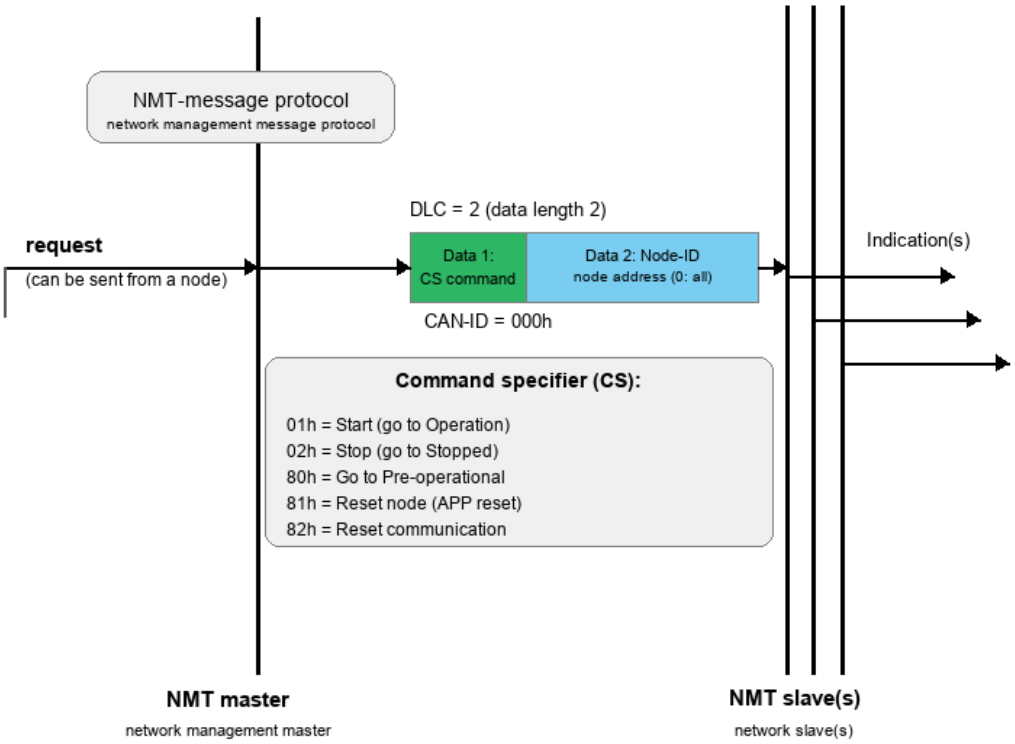


Figure B.7 NMT command specifier (CS)

2. Service Data Object (SDO): SDO is mainly used by the master to configure parameters of slaves. Service confirmation is its greatest feature: each message generates a response to ensure accurate transmission. In a CANopen system, the slave usually acts as the SDO server and the master as the client. Through the index and subindex, the client can access objects in the server's object dictionary. Thus the master can access any object dictionary entry of a slave, and SDO can transfer data of any length (data longer than 4 bytes is split into multiple messages).

The expedited SDO protocol is illustrated below.

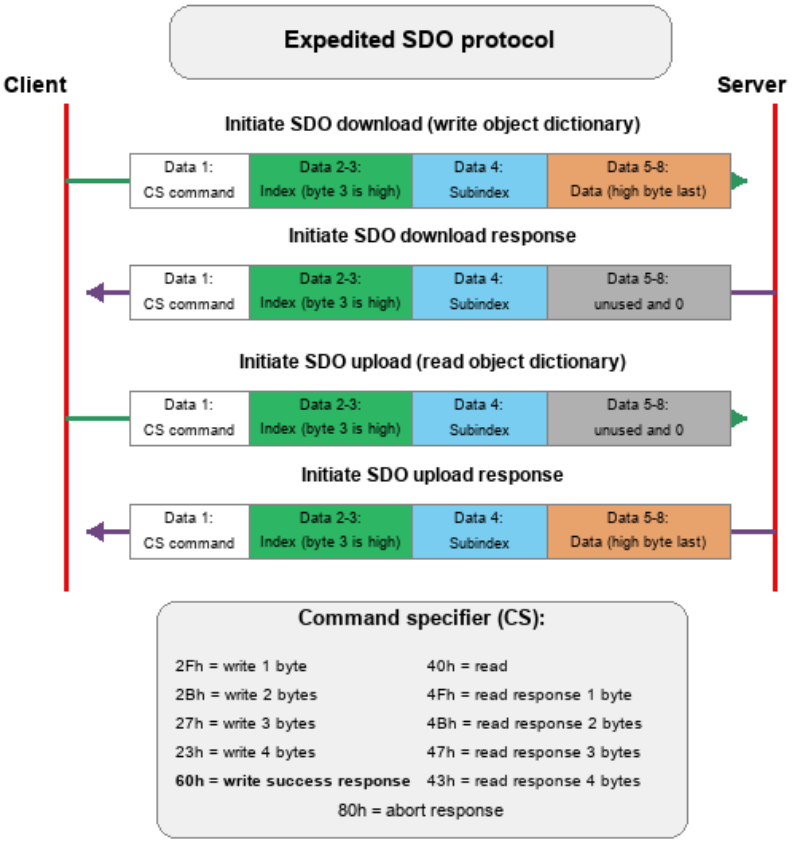


Figure B.8 Expedited SDO protocol

3. Process Data Object (PDO): PDO is used to transmit real-time data and follows the producer-consumer model shown in Figure B.9. The data length is limited to 1~8 bytes. A PDO communication object has the following characteristics:

- PDO communication has no protocol overhead; the PDO data content is defined by its CAN-ID (also called COB-ID).
- Each PDO is described by two objects in the object dictionary: PDO communication parameters (COB-ID, transmission type, event timer) and PDO mapping parameters (a list of mapped objects with data lengths in bits). Both producer and consumer must know the mapping parameters to interpret PDO content correctly.
- PDO message content is predefined; if variable PDO mapping is supported, the PDO can be configured via SDO.
- PDO can use synchronous transmission (triggered by receiving a SYNC object, either acyclic or cyclic; 1~240 sync objects can trigger it) or asynchronous transmission (triggered by a specific event: either a remote frame with the same COB-ID, or an object-specific event defined in the device profile such as timer-based or data-change transmission).

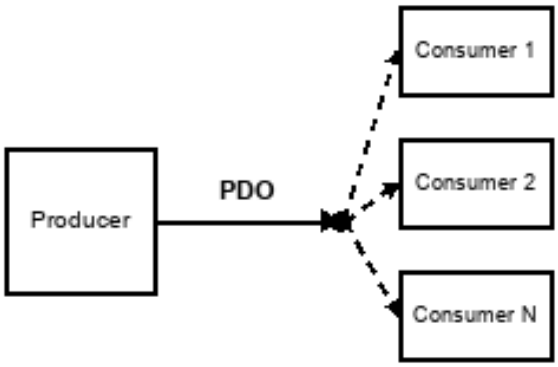


Figure B.9 Producer-consumer model

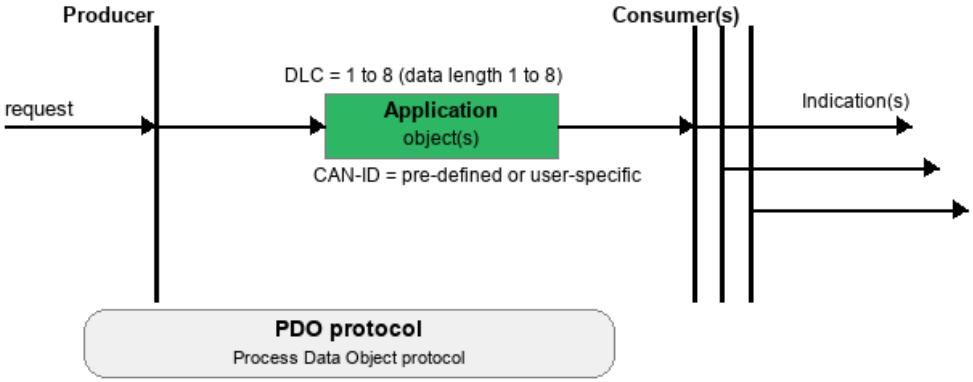


Figure B.10 PDO protocol communication

4. Predefined or Special-Function Objects: These provide specific functions for CANopen devices and ease master management of slaves. COB-IDs are predefined for special functions, mainly: Sync (SYNC) for network-wide synchronous transmission with high priority and short transmission time; Time Stamp for a common time reference; Emergency for internal device errors; Node/Life Guarding for the master to obtain slave status and vice versa; and Boot-up sent by a slave after initialization to enter the pre-operational state.

B.5 CANopen Network Configuration

The CANopen specification text DS305 defines a network configuration protocol, LSS (Layer Setting Service), through which a CANopen module with LSS master function can query or modify certain parameters of a CANopen module with LSS slave function over the CAN bus. Using LSS, the following parameters can be queried or modified:

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