A black square with white text

Description automatically generated

PILOT ICAN

Accelerator controller

ICAN-4404

4-channel isolated analog output module

A blue and grey electronic device

Description automatically generated

# **1. ICAN-4404** Feature Brief

The ICAN-4404 is an analog output module that can output four analog signals simultaneously with an internal 12-bit resolution DAC. The analog output signal can be configured as voltage signal output or current signal output by software, the voltage signal output range is 0~10V, and the current signal can be selected as 0~20mA or 4~20mA output. The module also has four digital input channels that can acquire level signals or switch contact signals to provide matching output functions for analog outputs.

The appearance of the ICAN-4404 module is shown in Figure 1.1.



Figure 1.1 Schematic diagram of ICAN-4404 appearance

Main technical indicators

## 1.1.1 Analog output

* Number of output channels: 4 channels;
* Output type: voltage output: 0~10V, current output: 0~20mA or 4~20mA;
* DAC resolution: 12 bits;
* Output accuracy: V: ±0.2%;I：±0.4%；；
* Output slope: configurable, voltage output: 0.0625V/s~1000V/s;Current output: 0.125mA/s~2000 mA/s;
* With safe start output and emergency output function, each channel is independently configured;
* 4 channels with synchronous output and sequential output;

## 1.1.2 Digital input

* Number of input channels: 4 ways;
* Input type: switch contact signal or level signal;
* Input range:

High (Figure 1): +3.5 V~+30V; Low (Digital 0): ≤+1V;

* Each DI channel can be configured for AO emergency output matching input function

## 1.1.3 System Parameters

* CPU: 32-bit RISC ARM;
* Operating system: real-time operating system;
* Isolation withstand voltage: 2500 VDC;
* Power supply voltage: +10~+30VDC, reverse power supply protection;
* Operating temperature range: -40°C~+85°C;
* Plastic housing, standard DIN rail mounting;
* Communication interface: isolated 2500 VDC, ESD, overvoltage, overcurrent protection;

## 1.2 Block diagram

The block diagram of the ICAN-4404 module is shown in Figure 1.2. The module is mainly composed of power supply, isolation circuit, D/A conversion circuit, digital input circuit, CAN isolated communication interface and MCU. The microcontroller of the module adopts 32-bit RISC ARM chip, which has very fast data processing capabilities and adopts a watchdog circuit, which can restart the system in case of accidents, making the system more stable and reliable, and can be used in high-performance and high-speed application environments.

ICAN-4404 is designed for industrial applications, and adopts photoelectric isolation between the internal input and output unit and the control unit, which greatly reduces the impact of industrial site interference on the normal operation of the module, so that the module has good reliability. Isolated

RS-485 communication interface, can avoid the influence of industrial field signals on the microcontroller communication interface, and has ESD, overvoltage, overcurrent protection.

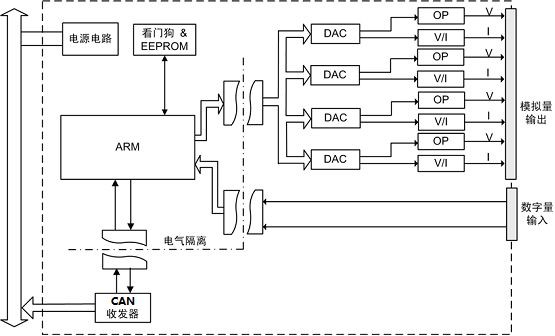


fig 1.2 ICAN-4404 Block diagram

## 1.3 Terminal Information

### **1.3.1** Terminal arrangement

The ICAN-4404 has a total of 26 terminals, and the arrangement of terminals on the housing is shown in Figure 1.3



fig 1.3 ICAN-4404 Terminal arrangement

### **1.3.2** Terminal description

The terminal definitions for ICAN-4404 are described as follows:

* GND, +VIN is the power input of the module, GND is connected to the negative terminal of the power supply, and +VIN is connected to the positive terminal of the power supply.
* EARTH is the earth terminal of the module, and connecting this terminal to earth can improve ESD protection performance.
* CFG is the default communication parameter hardware enable terminal of the module, when this terminal is grounded, the module will be initialized with default communication parameters, and the communication parameters are configurable.
* CANND, CANH, CANL is the isolated CAN interface terminal, and CANGND is the isolated ground of the communication interface

The CANN is connected to the H terminal of the CAN transceiver, and the CALL is connected to the L terminal of the CAN transceiver.

* DGND is the digital input port ground, and the analog output port ground is separated by a magnetic bead to avoid digital-analog interference.

The DGND, the module's power ground GND, and 485GND are galvanically isolated up to 2500 VDC.

* DI0~DI3 is the 4-channel digital input channel terminal of the module.
* VOUT0~VOUT3 is the voltage mode output port of AO0~AO3 channel;
* AGND is the analog output port ground;

IOUT0~IOUT3 is the current mode output port of AO0~AO3 channel, and adopts current sinking mode.

* +VEXT is an internal 12V output terminal, which can be used for internal power supply for current output.
* BANDRATE, ADDRX10, ADDRX1 DIP switches that set the baud rate and address for the module.

## 1.4 Electrical parameters

Unless otherwise specified, the parameters listed in Table 1.1 Electrical Parameters refer to values at Tamb=25°C.

Table 1.1 Electrical parameters

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| parameter | Parameter | minimum  Min. | Typical value  Typ. | maximum  Max. |  | unit  Unit |
| Analog output | Analog Output | |  |  |  | |
| DAC resolution |  |  | 12 |  | Bits | |
| precision | Accuracy | |  |  |  | |
| Voltage output | Voltage Output |  | ±0.1 | ±0.2 | %FSR | |
| Current output | Current Output |  | ±0.2 | ±0.4 | %FSR | |
| Output slope | Output Slope Rate | - 0.0625 |  | 1000 | V/s | |
|  |  | 0.125 |  | 2000 | mA/s | |
| Zero point temperature drift | Zero Drift | |  |  |  | |
| Voltage output | Voltage Output |  | ±30 | ±45 | uV/℃ | |
| Current output | Current Output |  | ±0.2 | ±0.4 | uV/℃ | |
| Full-scale temperature coefficient | Span temperature  coefficient |  | ±25 | ±45 | ppm/  ℃ | |
| Voltage output load | Voltage Output Load |  |  | 30 | mA | |
| Current output load voltage | Current Output Load Voltage | |  |  |  | |
| Internal 12V power supply |  |  |  | 8 | V | |
| External power supply |  |  |  | 30 | V | |
| Digital input | Digital Input | |  |  |  | |
| Logic low | Logic level 0 |  |  | 1 | V | |
| Logic high | Logic level 1 | 3.5 |  | 30 | V | |
| Isolation voltage | Isolation Voltage |  | 2500 |  | Vdc | |
| Supply voltage | Power Supply | 10 |  | 30 | V | |
| power consumption | Power Consumption |  | 2 | 4 | W | |

## 1.5 CAN baud rate and MAC ID settings

When using ICAN series function modules, you first need to set the baud rate of the CAN of the module and the MAC ID address of the module. The CAN baud rate and MAC ID of the module are set via a DIP switch.



Figure 1.4 ICAN-4404 DIP switch

As shown in Figure 1.4, the CAN baud rate has one dial control and the module MAC ID has two dial controls. The baud rate switch settings are shown in Table 1.1.

Table 1.1 Baud rate switch settings

|  |  |
| --- | --- |
| 0 | 1000Kbps |
| 1 | 800Kbps |
| 2 | 500Kbps |
| 3 | 250Kbps |
| 4 | 125Kbps |
| 5 | 50Kbps |
| 6 | 20Kbps |
| 7 | 10Kbps |

Note: When the DIP switch exceeds 7 , the CAN baud rate of the module automatically defaults to 1000Kbps.

The MAC ID setting of the module is determined by the two dials ADDRX10 and ADDRX1, when the module ADDRX10 dial points to 1, ADDRX1 If the dial points to 5, the MAC ID of the module is 1X10+5=15. Note that the MAC ID of the module cannot be 0 or exceed 127, when the module set address exceeds 127 or is 0, the module MAC

The ID defaults to 1.

### **1.6** Signal indicator

ICAN series modules have 3 LEDs, PWR is Power LED (green) and Working Status LED RUN, ERR. The PWR light is on, indicating that the ICAN module is powered normally. RUN is a green indicator and ERR is a red indicator, which can be seen from the housing panel to indicate the working status of the module. As defined in the CANopen protocol specification document DS303-3, two LEDs are used in the ICAN module to indicate the state of the current module, as shown in Table X 1.3X.

The various states indicated by its status indicator indicate the meaning of Table 1.4, Table 1.5, Table 1.6 below.

Table 1.3 Normal Functional Status RUN ERR LED Status

|  |  |
| --- | --- |
| LED name | color |
| Run indicator (RUN). | green |
| Error Indicator (ERR). | red |

Table 1.4 LED light status description

|  |  |
| --- | --- |
| LED status | Description of the phenomenon |
| LED on | Always on |
| Dark (LED off). | Always dark |
| LED flickering | Light and dark time is equal to the length, the frequency is about 10Hz: bright about 50ms, dark about 50ms |
| Blinking (LED blinking). | The light and dark times are equal in length, the frequency is about 2.5Hz, the light is about 200ms, and the dark is about 200ms |
| Flash | A short flash (about 200ms) is followed by a long period of darkness (about 1000ms). |
| LED double flash | Two short flashes of light ( about 200ms) separated by a dark of about 200ms.  This sequence ends with a long period of darkness (about 1000ms). |
| LED triple flash | Three short flashes of light ( about 200ms) separated by about 200ms of darkness in between . This sequence ends with a long period of darkness (about 1000ms). |

Table 1.5 Error Status Indicator (ERR) description

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| numbering | ERROR LED | state | description | class |
| 1 | Dark | There are no errors | The device is active | compulsion |
| 2 | Flash | Alert value reached | At least one error counter of the CAN controller has arrived or exceeded  Alert value (too many error frames) | compulsion |
| 3 | Flickering | Auto baud rate/  LSS | Automatic baud rate checks are in progress  Test or conduct LSS services (and RUN  LEDs blink alternately | Optional |
| 4 | Flash twice | Error control events | A protection event (NMT slave or NMT master) or heartbeat event (heartbeat consumer) occurs | compulsion |
| 5 | Flash three times | Sync error | The SYNC packet has not been received beyond the configured communication cycle interval (see Object Dictionary entry.)  0x1006） | Conditional, enforced if object 0x1006 is supported |
| 6 | Bright | The bus is off | The CAN controller bus is off | compulsion |

Table 1.5 describes the operational status indicators (RNUs).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| numbering | EUN LED | state | description | class |
| 1 | Flickering | Automatic baud rate/LSS | LSS service is in progress | Optional |
| 2 | Flash | Stop it | The device is in a stopped state | compulsion |
| 3 | Flickering | Pre-action | The device is in a pre-operational state | compulsion |
| 4 | Bright | Job | The device is active | compulsion |
| 5 | Dark | fault | Check that the module reset pin and the power supply are connected correctly |  |

## 1.7 Connection of CAN bus

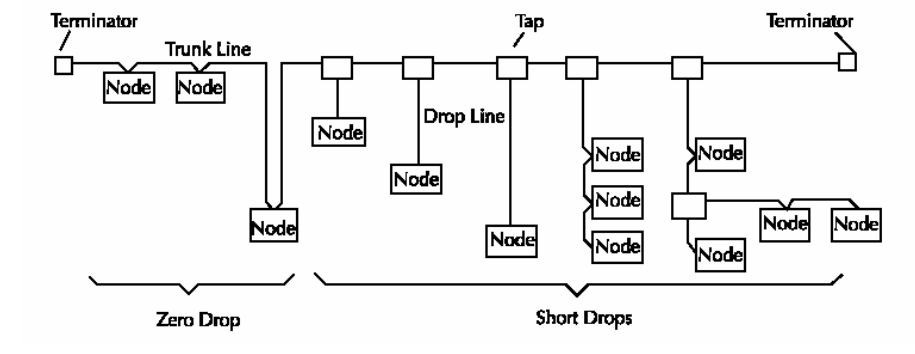


Figure 1.5 CAN network topology

CAN networks are bus-based topologies, and it is recommended to keep the length of branch lines as small as possible when wiring the network. A termination resistor is required at the end of the trunk of the CAN network. The trunk length of a CAN network is determined by the data transfer rate and the type of cable used. The cable distance between any two points in the cable system is not allowed to exceed the maximum cable distance allowed by the baud rate. The cable distance between two points is the sum of the length of the trunk and feeder cables between the two points. Feeder length refers to the maximum distance between individual transceivers from the trunk terminal to the node on the feeder. This distance includes feeder cables that may be permanently attached to the device. The total length of the allowed feeder on the network depends on the data transfer rate. When determining the length of trunk lines, the length and number of branch lines, it is necessary to refer to the relationship between the baud rate of CAN network communication and the communication distance, as shown in Table 1.3.

table 1.3 CAN The relationship between network communication baud rate and communication distance

|  |  |
| --- | --- |
| CAN baud rate | Maximum communication distance |
| 1000Kbps | 40m |
| 800Kbps | 100m |
| 500Kbps | 130m |
| 250Kbps | 270m |
| 125Kbps | 530m |
| 50Kbps | 1300m |
| 20Kbps | 3300m |
| 10Kbps | 6700m |

Note: CAN communication lines can use twisted pair, shielded twisted pair. If the communication distance exceeds 1KM, the cross-sectional area of the line should be greater than Φ1.0mm2, and the specific specifications should be determined according to the distance, and the routine is to increase appropriately with the lengthening of the distance.

The CAN-OPEN network requires termination resistors to be installed at both ends of the trunk, and the requirements for resistance are:

* 120 ohms;
* 1% metal film;
* 1/4 watt.

Note: The termination resistor should only be installed at both ends of the trunk, not at the end of the branch.

### **1.8** Connection of the power supply and communication cable of the module

When wiring, pay attention to the power supply and CAN communication line of the module: The +VS pin of the module is connected to the positive polarity end of the input power supply, and the GND pin is connected to the negative polarity end of the input power supply, so as to avoid the polarity error of the power supply connection when connecting. When multiple modules are connected to the same supply, all +VS pins are connected to the positive terminal of the supply and the GND pin is connected to the negative terminal of the supply. When the CAN communication line is connected, all module CAN\_L ends on the network must be connected to the same CAN\_L signal line, and all module CAN\_H ends must be connected to the same signal line, otherwise it will cause network communication abnormalities. This is shown in Figure 1.6.

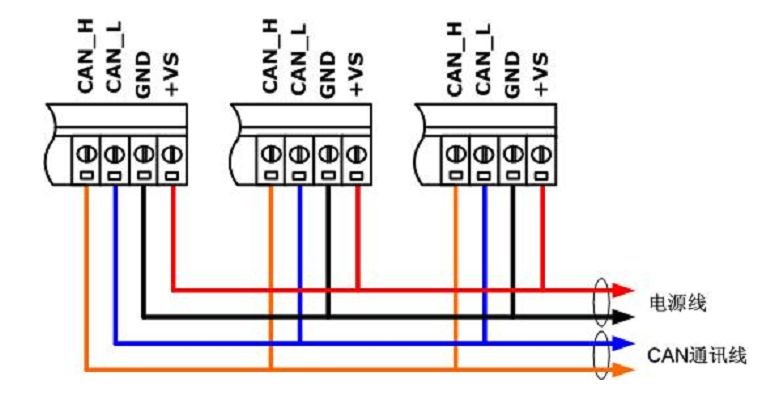


Figure 1.6 Connection of power supply and communication cables

### **1.9** Mechanical Specifications

#### 1.9.1 Mechanical dimensions

The ICAN series of data acquisition modules are housed in a plastic housing and have dimensions as shown in Figure 1.5.

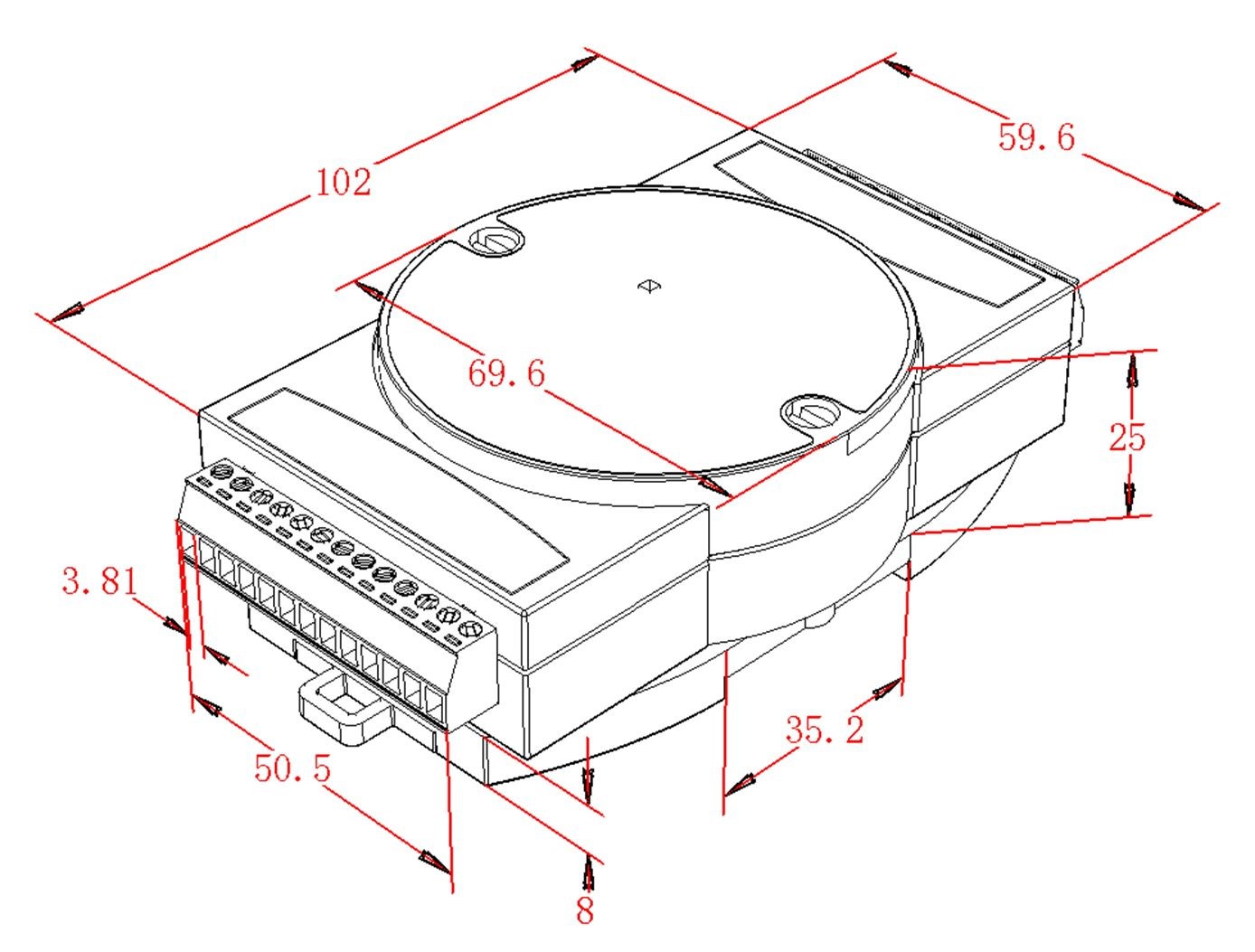


Figure 1.5 Schematic diagram of mechanical dimensions

#### 1.9.2 Installation method

ICAN series data module housings with rail baseplates, as shown in Figure 1.6, can be mounted directly on standard DIN rails (35mm wide D-rails), or other easy installation options are available to the user.

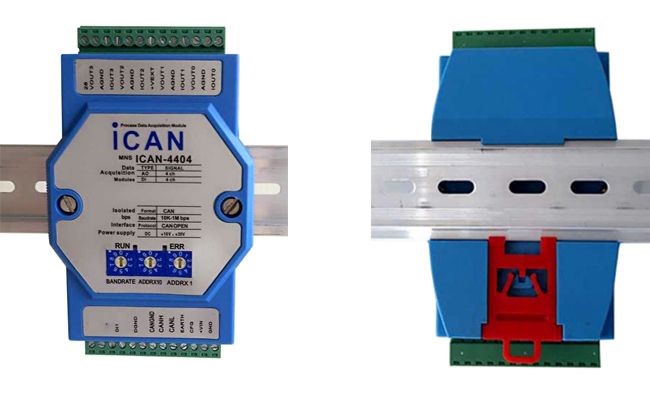


Figure 1.6 Schematic diagram of the rail base plate

When installing, first lock the ICAN module with the rail bottom plate, hook the rail bottom plate to the upper edge of the rail, and then pull the red card holder on the bottom plate down, attach the module bottom plate to the guide rail, and release the red card holder, that is, install the module on the rail, pictured

1.7 Schematic diagram of the installation process.

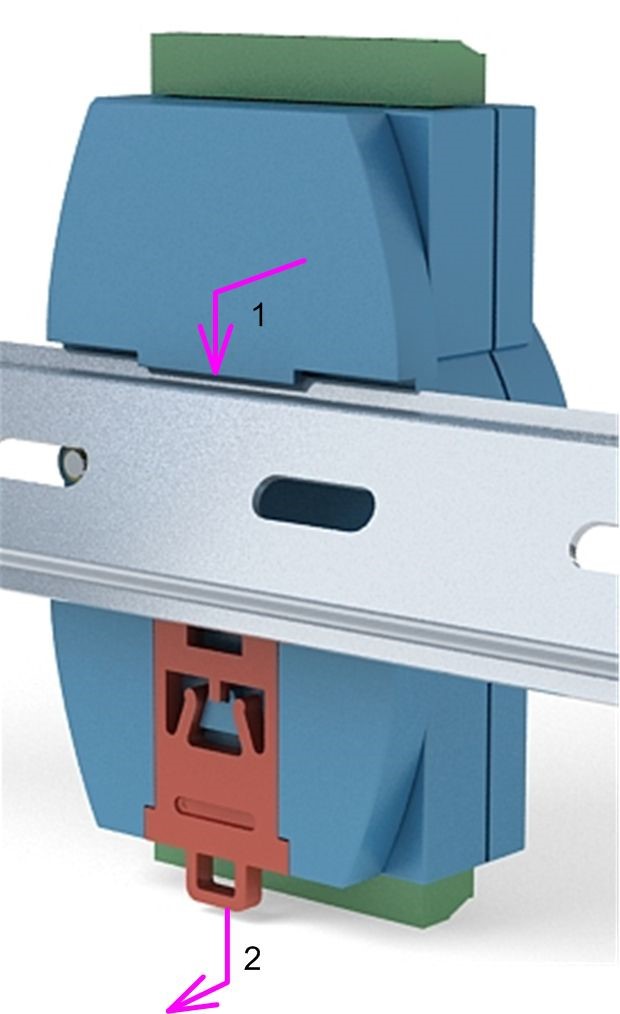


Figure 1.7 Installation schematic

## 2. Analog output function

### **2.1** Analog output

ICAN-4404 module has 4 analog outputs, each channel can be independently selected as voltage output or current output mode, the output range of voltage signal is 0~10V, and the current signal range can be selected as 0~20mA or 4~20mA 。

### **2.2** Output principle

The ICAN-4404 module enables analog output control via a digital-to-analog converter.

Digital-to-analog converter (DAC): Used to convert digital data into analog voltage or current signals, generally known as Digital/Analog Converter, digital-to-analog converter. The description of DAC conversion accuracy is usually expressed in bits (bits). The conversion accuracy of a DAC is closely related to the accuracy of the system output.

In the analog signal output system, in order to ensure the correctness of the analog output signal and the accuracy of the system, the analog signal output by the DAC needs to be conditioned. The circuit that completes this part of the conditioning function is generally called the "back-end circuit". The back-end circuit usually completes smooth filtering of the signal, adjustment of the signal amplitude range (such as adjustment of signal gain), conversion of signal type (I/V, V/I conversion), and so on.

The basic structure of the ICAN-4404 back-end circuit is shown in Figure 2.1.

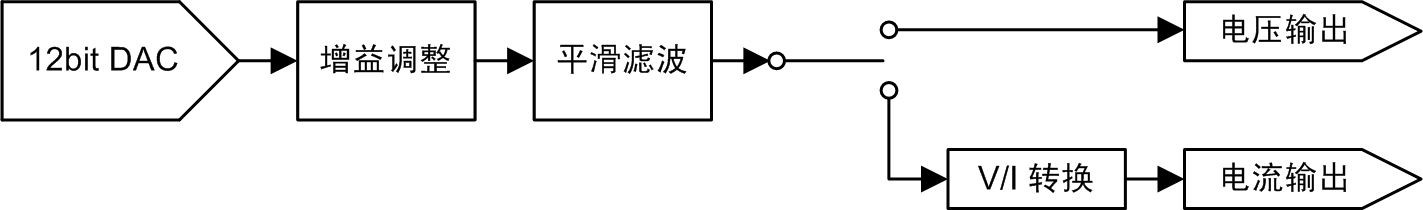


Figure 2.1 Analog output back-end conditioning circuit

Its back-end circuit basically consists of a DAC, a gain adjustment circuit, a smoothing filter, and a signal conversion circuit. The gain adjustment circuit adjusts the amplitude of the DAC output signal to a more appropriate voltage as needed, the smoothing filter filters filter the DAC output signal, and the V/I conversion circuit converts the voltage signal to a current signal.

### **2.3** Output wiring

The ICAN-4404 has 4 analog output channels, which can be selected as voltage output mode or current output mode. The voltage output wiring is shown in Figure 2.2.

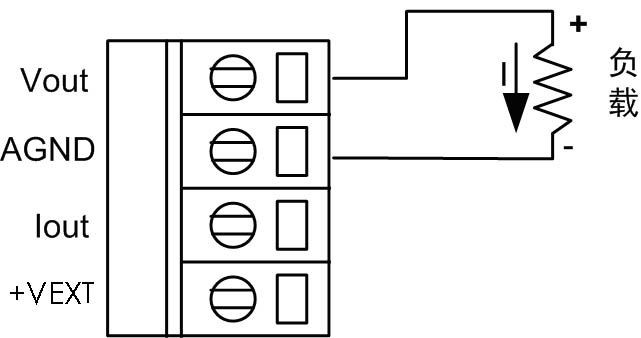


Figure 2.2 Schematic wiring diagram of analog output voltage mode

With voltage output, the maximum load current per channel is 20mA, and when the output is overcurrent (e.g. output short circuit), all AO channel outputs will be abnormal and may cause module damage.

When the analog output selects the current output, it can use the internal power supply mode or the external power supply mode, such as the wiring method

Figure 2.3 shows.

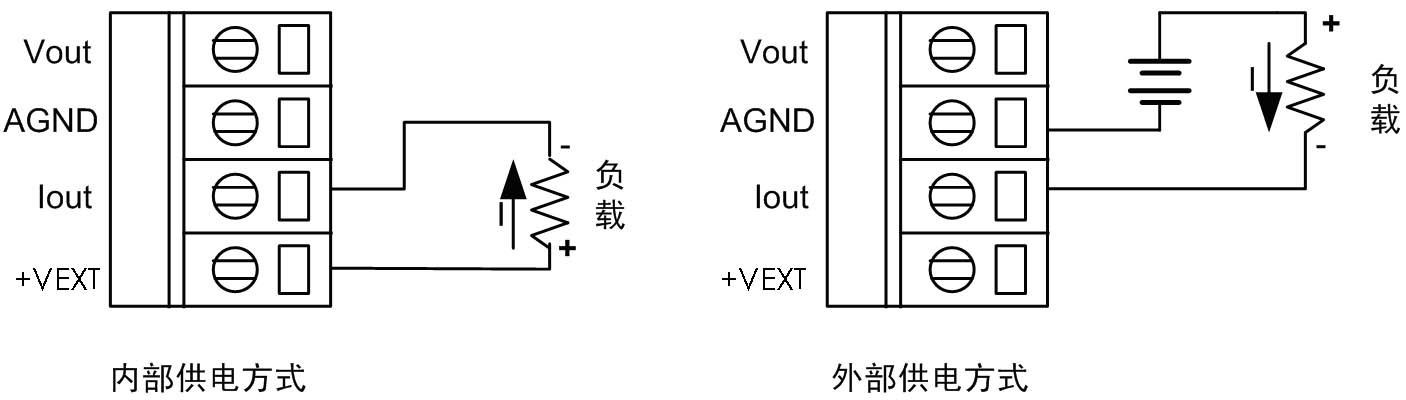


Figure 2.3 Schematic diagram of analog output current mode wiring

With an external power supply, the maximum load voltage cannot exceed +24V.

Each AO channel can only use one output method at the same time, when choosing the voltage output mode, it should be ensured that there is no load connected to the current output terminal of the same channel; when the current output mode is selected, it should also be ensured that there is no load connected to the voltage output terminal of the same channel; otherwise, it may cause abnormal output and even damage the module.

Note: Whether the analog output channel of the module is configured as voltage output mode or current output mode, the maximum output load current of a single channel is 20mA must be in the range of -40~70°C, when the working ambient temperature is within the range of 71~85°C, the total load current can only be 60% of the rated current, i.e. less than 24mA. For the current output method, an external power supply method is recommended.

### **2.4** Output value calculation

The ICAN-4404 module uses a 12-bit resolution DAC to control the analog output with an output of 0x000 zero and a full-scale value of 0xFFF. Depending on the output type and range selected, convert the analog value to be output to a decimal 16 value and write

AO port register, the module will output the corresponding analog signal.

*X*

For example, to output the voltage of X(V), calculate 4095, and then convert the result to base 16.

### 10

*X*

To output the voltage of X(mA), calculate 4095 and convert the result to base 16.

### 20

The configuration software configuration command for the channel output type is implemented by writing configuration code, and the correspondence between the configuration code and the output range is such as Error! Reference source not found. shown.

Table 2.1 AO Channel Output Range Settings

|  |  |
| --- | --- |
| Quantum code | Output range |
| 00 | 0～20mA |
| 01 | 4～20mA |
| 02 | 0～10v |

In the 4~20mA output range, the command output is less than 4mA, and it will be output at 4mA.

## 3. Digital input function

The ICAN-4404 also has four digital input channels that can be set as normal DI input or AO output matched input.

### **3.1** Normal **DI** input mode

The DI channels of the ICAN-4404 module can be used to acquire voltage or passive contact digital signals, and the input signal logic states are defined as shown in Table 3.1.

Table 3.1 Input signal definitions

|  |  |  |
| --- | --- | --- |
| Input signal type | | Signal definition |
| Voltage-type digital input signal | High-level signal | State 1, voltage range: +3.5 V~+30V |
| Low level signal | Status 0, voltage range: ≤+1V |
| Passive contact-type digital input signal | Open contact signal | State 1 |
| Close the contact signal | Status 0 |

A schematic of the ICAN-4404 digital input port is shown in Figure 3.1. The left side of the figure is the external wiring, when the external input is a level signal, when the voltage of the input signal is less than 1V, the optocoupler is on, point A outputs a low level, and the logic state is 0; when the voltage of the input signal is greater than 3.5V and less than 30V, the optocoupler cut-off, point A outputs a high level, and the logic state is 1.

When the module switches the contact signal, when the switch is closed, the optocoupler is on, and the logic state is 0; similarly, when the switch is off, the optocoupler is off and the logic state is 1.

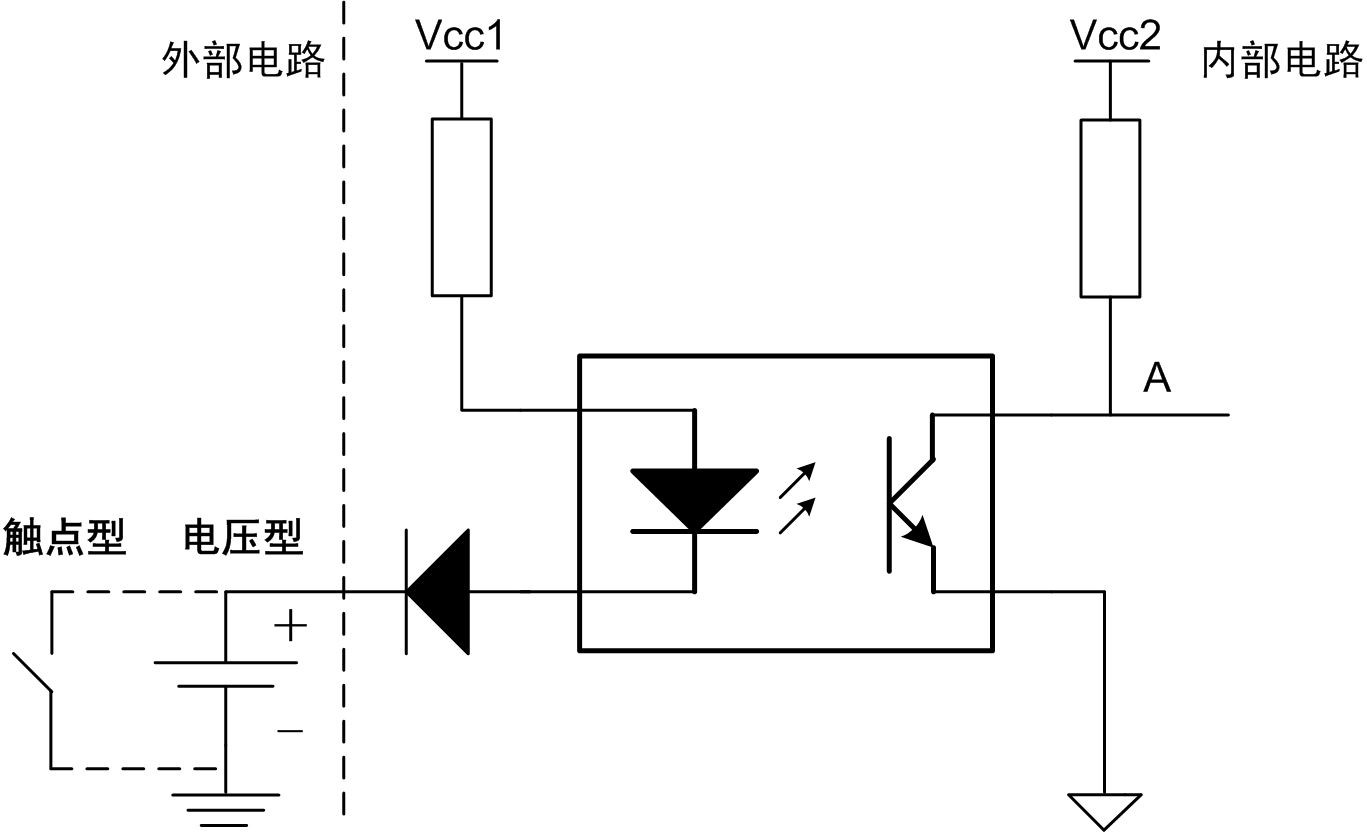


Figure 3.1 Schematic diagram of the digital input principle

### **3.2 AO** output matches input mode

The ICAN-4404's four DI channels can be independently configured as matching inputs for the emergency output function of the AO channel, and the input matching level can be configured by configuration software. When the DI channel is configured as a matched input mode and the DI port input level is consistent with the set matching level, the corresponding AO channel is output with the set emergency output value.

### **3.3** Input wiring method

The DI channel of the ICAN-4404 can read the current input value of the DI port from the DI registers regardless of which mode of operation is chosen, and the wiring pattern for all three modes is the same, as shown in Figure 3.2. Pay attention to the polarity of the signal when wiring the voltage digital signal to avoid reverse connection.

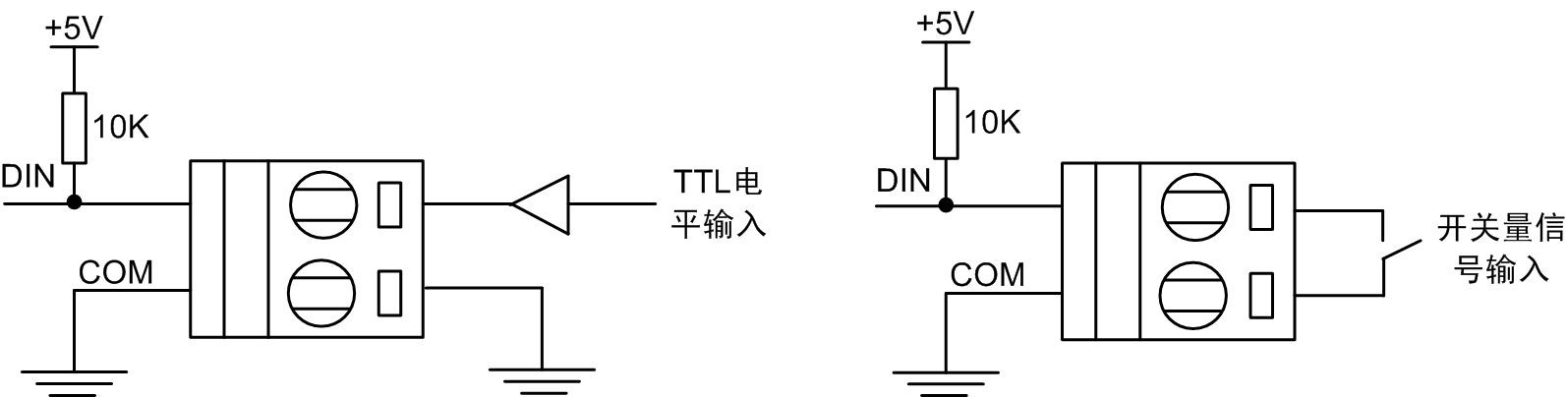


Figure 3.2 Schematic diagram of the digital input wiring method

## 4. Parameter configuration for the module

The Canopen protocol is one of the standards defined by CAN-in-Automation (CA) and gained widespread acceptance shortly after its release. Especially in Europe, the Can open protocol is considered the leading standard in CAN-based industrial systems. Most important device types, such as digital and analog I/O modules, drive devices, operating devices, controllers, programmable controllers, or encoders, are described in a protocol called "device description"; "device description" defines different types of standard devices and their corresponding functions. With the support of the Canopen protocol, devices from different manufacturers can be configured via the bus.

### **4.1** Description of communication parameters

The list of object dictionaries is shown in Appendix A, except for the parameters defined by the device manufacturer, which are standard parameters in the Canopen protocol, please refer to the protocol text for details. Device vendor defined parameters start at 0x2000, as shown in Table 4.1.

Table 4.1 Table of device vendor defined parameters

|  |  |  |  |
| --- | --- | --- | --- |
| index | Subindex | The parameter name | illustrate |
| 0x2000 | 0 | Number of Entries | Number of parameters |
| 1 | Channel Enable | Channel enable |
| 0x2001 | 0 | Number of Entries | Number of parameters |
| 1 | Channel Range1 | Channel type |
| 2 | Channel Range2 | Channel type |
| 3 | Channel Range3 | Channel type |
| 4 | Channel Range4 | Channel type |

### **4.2** Parameter configuration

Parameters in the object dictionary can be configured by the user if their properties are writable.

**4.2.1** Use dedicated configuration software

There are Canopen master station cards and supporting Canopen network configuration software sold on the market, the system is generally more powerful, can not only configure node parameters, but also monitor the entire network, easy to use, but the price is more expensive.

**4.2.2** Use of General Software

If the user does not have a dedicated Canopen network configuration system, the module parameters can also be configured in a simple way, as follows.

1. , purchase a simple CAN communication interface card, as long as it supports CAN2.0, install it on the computer, and run the test software provided by the interface card manufacturer, as shown in the following figure;
2. , set the communication rate and other communication parameters;
3. , configuration parameters: examples are as follows. a, read the object dictionary: as shown in Figure 4.1.

0x601: SDO is identified as 0x600+Anode (Module Identification ID);

40001805: Read the item with index 0x1800 and subindex 05 , which is the event time Event Time; 40001803: Read the item with index 0x1800 and subindex 03 , that is, the constraint time Inhibit Time.

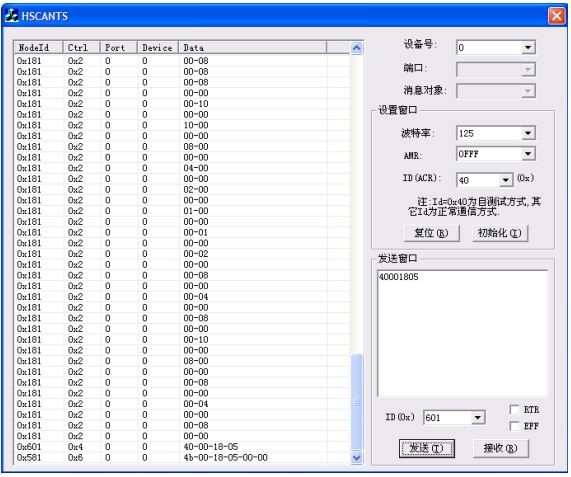


Figure 4.1 Reading the Object Dictionary

b. Write object dictionary: as shown in Figure 4.2.

0x601: SDO is identified as 0x600+Anode; change the constraint time to 0, and the SDO command is: 220018030000; Change the event time to 1000 milliseconds, or 0x3E8, and the SDO command is: 2200180503E8;

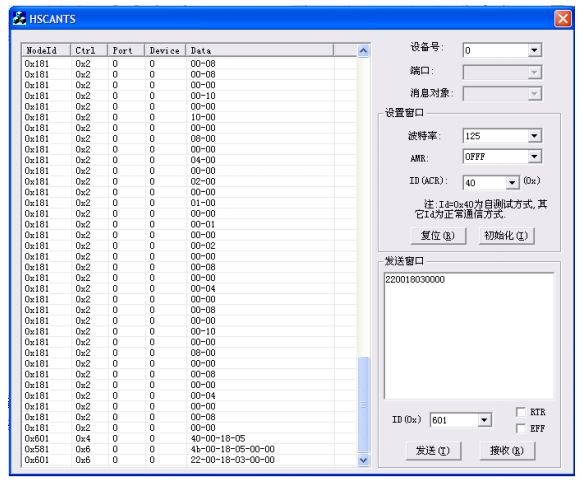


Figure 4.2 Write Object Dictionary

# **5. ICAN-4404'**s **Canopen** protocol and its applications

CAN only defines Layer 1 and Layer 2 (see ISO11898 Standard), that is, only the physical layer and the data link layer, not the application layer, so an application layer protocol is required to implement the interconnect, as shown in Figure 5.1. CAL (CAN Application Layer) protocol is currently one of the CAN-based high-level communication protocols, which is operated by CA (CAN in Automation) is responsible for the management, development and promotion. CALs provide all network management services and messaging protocols, but do not define the content of CMS objects, Canopen is based on CALs and uses CALs A subset of communication and service protocols, providing an implementation scheme for distributed control systems. After many modifications to the Canopen protocol specification, the stability, real-time and anti-interference of the Canopen protocol have been further improved, and CA has continuously introduced device sub-protocols in various industries to make Canopen Protocols are developing faster in various industries.

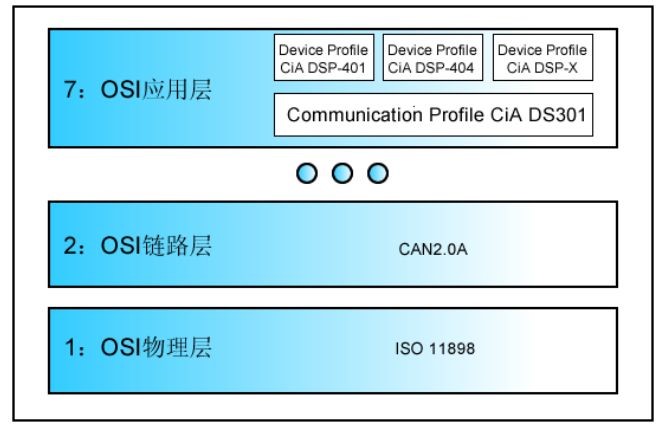


Figure 5.1 Block diagram of the location of CAN, Canopen standards in the OSI network model

## 5.1 NMT object

NMT is suitable for CAN networks with Master/Slave architecture. For a Canopen network with a Master/Slave structure

The master node manages and controls the entire network, including each slave node, which can not only control the running status of each slave node, but also monitor each slave node, which plays a very important role in the quality and performance of network operation.

There can only be one Master node in a network, but there can be multiple slave nodes.

The commands supported by NMT are shown in Table 5.1 :

Table 5.1 NMT Command Table

|  |  |  |
| --- | --- | --- |
| NMT command | value | exegesis |
| Start Remote Node | 1 | Start the node (enter Operation). |
| Stop Remote Node | 2 | Stop the node |
| Entry Are\_Operation | 128 | Enter the Are\_Operation |
| Reset Node | 129 | Reset the node |
| Reset Communication | 130 | Reset communication parameters |

A node can select one of four states, as shown in Table 5.2.

Table 5.2 Node Status Table

|  |  |
| --- | --- |
| state | Status value |
| Industrialization(Boot) | 0 |
| Noncooperation | 127 |
| Operation | 5 |
| Scoped | 4 |

After the node starts, it defaults to the pre-run state Noncooperation, and then the master node controls it to enter the running state, but the user can also make it directly enter the running state after starting, without the need for the master node to control (see Module Startup for details).

## 5.2 SYNC object

The SYNC (Synchronization) object is mainly used to synchronize nodes on the network, that is, a node on the network periodically sends SYNC messages at regular intervals, mainly used to synchronize Dos.

The SYNC (Synchronization) object is based on the Producer/Consumer communication model. The SYNC message producer in the Can open network periodically sends SYNC messages, and all other nodes should send PDO immediately after receiving the SYNC message , the PDO validity time is determined by Synchronous windows. The SYNC message contains no data.

Communicate Cycle Time, the time interval of SYNC, in microseconds (0x1006).

Synchronous windows, the width of the synchronization window, in microseconds, which is the time interval between receiving a SYNC message and sending the PDO. In general, after receiving a SYNC message, the data must first be refreshed and then sent, and the time of this process cannot be greater than the width of the synchronization window. For RPDO, this time refers to the shortest time (0x1007) between RPDO outputs.

## 5.3 Time Stamp object

Network synchronization can also employ a Time Stamp object, which is based on the Producer/Consumer communication model. TIME production provides a time reference for TIME consumers on the web. A node can be either Time

Stamp producers, who can also be Time Stamp consumers (0x1012).

This module does not support.

**5.4 EMCY** Objects

The EMC (Emergency) object, or alert object, is mainly used to report the error state of the network.

## 5.5 Node Guard object

For CAN networks with a Master/Slave structure, the Master node uses the Node Guarding protocol to obtain the state of the slave node, while for non-Slave nodes The CAN network with the Master/Slave structure uses the Heartbeat protocol (or object) to obtain the state of the node and determine whether the node is working properly. A device either uses either the Node Guarding protocol or the Heartbeat protocol, and two protocols cannot be used on the same device at the same time.

The Node Guarding protocol is mainly used for master Guard slave nodes, which stipulates that NMT master nodes must use Pull way to communicate with the slave node, that is, the master node first sends the remote data request frame RTR to the slave node, slave After receiving the request, the node sends its current NMT state to the master node in the form of a data frame (using the Life Guarding protocol for slave node guarding Master node), to complete the secondary communication.

The NMT Master node sends Node Guarding messages (RTRs) periodically at Guard Time, and the NMT Slave node receives the Master periodically Life Guarding should be sent immediately after the message from the node if the NMT slave node is in Life

If no Node Guarding message is received from the Master node within Time, the Life Guarding Event is produced, that is, a Guard error occurs. Similarly, if the master node does not acknowledge sending RTR messages to the slave node (i.e The master did not receive data from the slave node during the life time time), or the resulting state is not the desired state (e.g. slave). The state of the node has changed), and a Node Guarding Event is generated, that is, a Guard error occurs.

The Node Guarding message format is shown in Table 5.3

Table 5.3 Node Guarding message format

|  |  |  |  |
| --- | --- | --- | --- |
| COB-ID | Data field length DLC | Data Data | Frame format |
| 0x700+NodeId | 0 | not | RTR |

NodeId: The node-ID of the slave node. The Life Guarding message format is shown in Table 5.4

Table 5.4 Life Guarding Message Format

|  |  |  |  |
| --- | --- | --- | --- |
| COB-ID | Data field length DLC | Data Data | Frame format |
| 0x700+NodeId | 1 | Status | Data frame |

NodeId: The Node-ID of the slave node.

Status: The NMT state of the slave node, as shown in Table 5.5:

Table 5.5 NMT Status for Slave Nodes

|  |  |
| --- | --- |
| Status value | NMT status |
| 4 | STOPPED |
| 5 | OPERATIONAL |
| 127 | PRE-OPERATIONAL |

The highest bit in Status (bit7) is the toggle bit, which should alternate with a starting value of 0. The Toggle bit is used to determine whether the received data is legitimate.

Node Guarding has two parameters that need to be set by the user:

Guard Time: In milliseconds (0x100C);

Life time factor: Product factor, Life Time = Guard Time\*Life time Factor, (0x100D);

If one of the two parameters is 0, the object is revoked or closed

## 5.6 Heartbeat objects

The Heartbeat object is based on the Producer/Consumer (Consume) communication model. Mainly used for Can open network management without NMT Master node. The Heartbeat producer periodically sends heartbeat messages, and its period is determined by the Producer Heartbeat Time parameter in the object dictionary (0x1017). ), the message format is shown in Table 5.6:

Table 5.6 Heartbeat message formats

|  |  |  |
| --- | --- | --- |
| COB-ID | Data field length DLC | Data Data |
| 0x700+NodeId | 1 | Status |

The status status is shown in Table 5.7 without the toggle bit.

Table 5.7 Status status values

|  |  |
| --- | --- |
| Status value | NMT status |
| 0 | BOOTUP |
| 4 | STOPPED |
| 5 | OPERATIONAL |
| 127 | PRE-OPERATIONAL |

Heartbeat Consumer In order to effectively monitor Heartbeat producers, Heartbeat Consumer needs to know the node ID and consumption time of each Heartbeat Producer ( Consumer Heartbeat Time), which is saved in the 0x1016 (data type Array) of the object dictionary. This allows Heartbeat consumers to monitor the status of each Heartbeat producer.

Producer Heartbeat Time: Units in milliseconds (0x1017);

Consumer Heartbeat Time: Units in milliseconds (0x1016);

When a node transitions from the INITIALISING state to the PRE-OPERATIONAL state, if the Heartbeat Producer Time is greater than zero, that is, the Heartbeat protocol starts The boot message is used as the first heartbeat message.

## 5.7 PDO objects

The system uses the default predefined set of connections, i.e. 4 TPDOs, 4 RPDOs. Each is explained below.

PDOs are used to transmit process data with high real-time requirements, such as analog or switching signals. For multihomed networks (where any node can send data to the network at any time), the PDO must have a higher priority. PDO can send multiple data items in an object dictionary in packages (up to 8 bytes) or in segments of more than 8 bytes.

PDO is based on the producer-consumer communication model, i.e. one PDO producer, one or more PDO consumers, and PDO consumers consume (receive data) directly without a response. The PDO that the PDO producer sends (sends PDO data to the network) is called a send PDO (TPDO), the same PDO A PDO received by a consumer (receiving PDO data from the network) is called a receive PDO (RPDO). The trigger mode (that is, the sending condition) of the PDO can be an event or a scheduled sending, etc., and the specific trigger method is determined by the PDO Communication Parameters.

（1），PDO Communication Parameters；

The communication parameters of the PDO determine the communication characteristics of the PDO, that is, how the PDO is triggered, the priority of the PDO, and the pair

Some time limits for PDO to transmit data, etc., the communication parameters of PDO are shown in Table 5.8:

table 5.8 PDO communication parameters

|  |  |  |  |
| --- | --- | --- | --- |
| index | Subindex | Communication parameters | illustrate |
| 1800h | 0 | Number of Entries | Number of parameters |
| 1 | COB-ID | PDO identification |
| 2 | Transmission Type | Transport type |
| 3 | Inhibit Time | TPDO minimum time |
| 4 | Reserved | retain |
| 5 | Event Time | The maximum time interval between TPDOs |

(a), COB-ID: The identity of the PDO communication object that determines the priority of its communication, the specific value of which is shown in Table 5.9.

table 5.9 COB-ID definition

|  |  |  |
| --- | --- | --- |
| Tag bit | Value | function |
| 31（MSB） | 0 | PDO exists or is valid |
| 1 | PDO does not exist or is invalid |
| 30 | 0 | RTR is allowed |
| 1 | RTR is not allowed |
| 29 | 0 | 11-bit ID (CAN2.0A). |
| 1 | 29-bit ID (CAN2.0B). |
| 28-11 | 0 | If bit29=0 |
| x | If bit29=1, bit 28-11 of COB-ID |
| 10-0（LSB） | x | The bit 10-0 of the COB-ID |

(b)，Transmission Type；

TPDO: divided into periodic and non-periodic sending, periodic sending refers to sending data after receiving a synchronization message, regardless of whether the PDO data changes, non-periodic sending refers to sending data when an event occurs, events include receiving a SYNC message, data changes, Event Time to or from RTR, etc. The value range of Transmission Type is 0-0xFF, and the meaning of each value is described in the Can Open protocol specification.

RPDO: RPDO messages are usually received, but the output depends on the occurrence of events, such as the receipt of the synchronous message SYNC or the change in the received data (compared to the previous time).

(c)，Inhibit Time；

For TPDO, if the time interval between two PDO is relatively short, it will increase the load of the CAN bus and cause problems such as bus blocking, so the time interval between the two PDOs can be set by the user, which is Inhibit Time, that is, the time between TPDO cannot be less than Inhibit Time.

For example, if the data sent by the TPDO is the result of an A/D transformation and the Transmission Type is the data change event, because the lowest bit of the A/D transformation result is usually constantly changing, there is no constraint time limit (Inhibit Time), the time interval between sending PDO is very short, and the bus will be blocked.

(d)，Event Time；

TPDO: As soon as the Event Time time is up, the PDO is sent immediately (that is, timed sending), regardless of whether the data has changed. The calculation of Event Time time begins after a PDO is sent. If Event Time=0, the trigger mode is canceled.

RPDO: If 254 or 255 is selected for Transmission Type, or if no PDO is received within the Event Time time PDO error.

|  |  |  |  |
| --- | --- | --- | --- |
| Trans.Type  ( transport type). | Whether to synchronize | Data requisition data acquisition and transmission (TPDO). |  |
| 0 | Aperiodic synchronization | Data is sampled every time a SYNC is received and sent if the data changes  PDO。 |
| 1-240 | Periodic synchronization | N SYNCs are received to sample the data once (N=1-240) and then the PDO is sent . |
| 241-251 |  | retain |
| 252 | synchronous | Data is collected when a SYNC is received and sent when a remote request frame is received  PDO。 （RTR） |
| 253 | asynchronous | Collecting data is carried out as a value and a PDO is sent when a remote request frame is received. （RTR） |
| 254 | asynchronous | Data is collected after a user-defined trigger event occurs and the PDO is sent immediately. |
| 255 | asynchronous | Data is collected after a trigger event defined by the device file and the PDO is sent immediately. |
| Trans.Type  ( transport type). | RPDO | | |
| 0 | Receives a PDO and flushes the output data when a SYNC is received and the data changes. | | |
| 1-240 | When the PDO is received, the output data is refreshed when N syncs (N=1-240) are received and the data changes. | | |
| 241-251 | Retain. | | |
| 252 | Retain. | | |
| 253 | Retain. | | |
| 254 | Receives the PDO and refreshes the output after a user-defined trigger event occurs. | | |
| 255 | Receives the PDO and flushes the output when a trigger event defined by the device file occurs. | | |

(2), PDO Mapping Parameters.

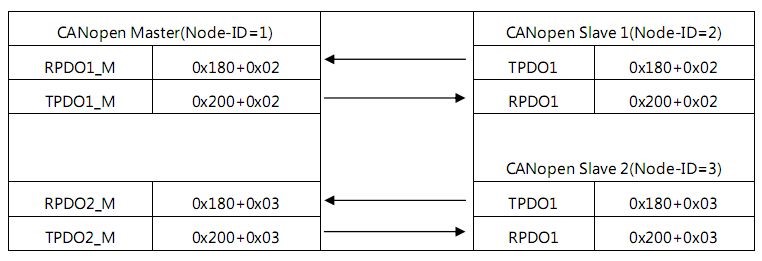
The PDO mapping parameter defines the data source location for the PDO, that is, the address of the process data that the PDO object will send, such as the address of an I/O port or the address of an A/D buffer. Its setting parameters include the index in the object dictionary, the number of mappings, the length of each mapping, and so on.

A PDO can only send up to 8 bytes of data at a time, that is, 64-bit data, so the number and length of mappings should meet this requirement, if the length of each mapping is 8 bits (0x08). ), the number of mappings can be 8, and if the length of each mapping is 16 bits (0x10), the maximum number of mappings is 4. (3), PDO Linking:

Since a PDO producer can have multiple PDO consumers, how to arrange the connection between the distribution producer and the consumer, this is the problem that the PDO connection solves. For Master/Slave network structures, PDO exchange data is done through the Master node, 4 TPDOs and 4 TPDOs in the predefined connection set RPDO is aimed at this network structure.

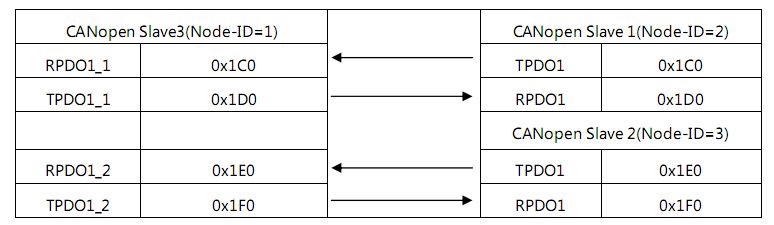
In general, for the Master/Slave network structure, PDO data is transmitted from the slave node to the master node, or from the master node to a certain node slave node, but sometimes PDO data needs to be transferred between two slave nodes, in which case the PDO data should be transferred through the master node completes, as shown in Figure 5.2, the Slave1 node sends data to the RPDO1\_M of the Master node via TPDO1, and then The master node sends the data to RPDO1 of slave2 through the TPDO2\_M to complete a PDO data transfer.

fig 5.2 Master node with Slave node PDO Data transmission



Data can also be exchanged directly between two slave nodes without going through the master node, in which case the node ID of the PDO producer (TPDO) must be the same as the COB-ID The node IDs of the PDO consumers (RPDOs) are identical and unique. The transfer process is shown in Figure 5.3.

fig 5.3 Slave node with Slave node PDO Data transmission



## 5.8 SDO Object

SDO (Service Data Objects) uses a client/server model and is mainly used to read and write object dictionaries for each node in the Can open network. The object dictionary is an array structure with many Entries, and the addressing methods of items use Index and Sub index. The data type of Index is UINT16, and the data type of Sub index is UINT8, that is, SDO can be addressed at most 65,536 items, with a maximum of 256 children per item. There are many data types of items in the object dictionary, including integer, string, array, structure, domain, etc., so the length of the data varies greatly, for SDO, some frames can be completed, and some need many frames to complete, so in

In the CA-DS305 protocol specification, Can Open defines three SDO subprotocols, as shown in Table 5.10:

table 5.10 SDO Sub-protocol

|  |  |  |
| --- | --- | --- |
| Protocol | Data length | illustrate |
| Expedited Transfer | 1-4 bytes | One frame (8 bytes). |
| Segmented Transfer | Any number of bytes | Multiple frames |
| Block Transfer | Any number of bytes | Multiple frames |

Read (Upload) Write (Download) is relative to the server side.

The read and write request of SDO is generally initiated by the SDO client (Client), and the SDO server responds to the request to complete an SDO read and write process. In order to avoid conflicts, SDO transmission generally requires two different COB-IDs, which are used for SDO transmission in different directions, and for SDO Server nodes, the default two SDOs SDOT, respectively

(0x580+Anode) and SDOR (0x600+Anode), SDOT is used to send SDO (Server->Client).SDOR is used to receive SDO (Client->Server). For SDO client nodes, the direction is reversed, and SDOT is used to receive SDO (Server->Client), SDOR Used to send SDO (Client->Server). An SDO generally has three parameters, which are described below.

1. The definition of COB-ID (Client->Server) is shown in Table 5.11.

table 5.11 COB-ID definition

|  |  |  |
| --- | --- | --- |
| Tag bit | Value | Function |
| 31（MSB） | 0 | SDO exists or is valid |
| 1 | SDO does not exist or is invalid |
| 30 | 0 | Reserved, defaults to 0 |
| 29 | 0 | 11-bit ID (CAN2.0A). |
| 1 | 29-bit ID (CAN2.0B). |
| 28-11 | 0 | If bit29=0 |
| x | If bit29=1, bit 28-11 of COB-ID |
| 10-0（LSB） | x | The bit 10-0 of the COB-ID |

COB-IDs range from 0x601—0x6FF, the corresponding entries in the object dictionary are 0x1280—0x12FF, and all items are read-write (RW).

1. The definition of COB-ID (Server->Client) is shown in Table 5.11.

COB-IDs range from 0x581—0x7FF, the corresponding item in the object dictionary is 0x1200—0x127F, and the properties of 0x1200 items are read-only ( RO), the properties of other items (0x1201-0x127F) are read-write (RW).

1. ，Node-ID（0x1-0x7F）：

For SDO Server, the Node-ID is the Node-ID of the SDO Client, which is optional; for the SDO Client, this parameter is optional Node-ID is the Node-ID of the SDO Server, which is required. The main function of this parameter is to determine the node ID in COB-ID, Node-ID. For the Can Open network with Master/Slave structure, there is generally only one SDO Client node and multiple SDO Server nodes, for ease of use SDO clients are generally placed on NMT master nodes, and SDO servers are generally placed on NMT slave nodes.

## 6. Disclaimer of Warranties

copyright

The product text and related software copyrights stated in this manual are owned by Quanzhou Guanhangda Electronic Technology Co., Ltd., and its property rights are absolutely protected by national laws, without the authorization of the company, other companies, units, agents and individuals shall not illegally use and copy, otherwise they will be severely punished by national laws.

Right to modify the Documentation

Quanzhou Guanhangda Electronic Technology Co., Ltd. reserves the right to modify this manual at any time without prior declaration.

# Appendix **A:** Object Dictionary

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| index | Subindex | Object name | data type | attribute | Default value | description |
| 1000 | 0 | Device type | Unsigned32 | const | 0x00060191 | Device type |
| 1001 | 0 | Error register | Unsigned8 | ro | 0 | The current error type |
| 1002 | 0 | Status register | Unsigned32 | ro | 0 |  |
| 1003 | 0 | Are-define Error field | Unsigned32 | ro |  |  |
|  |  |  |  |  |  |  |
| 1005 | 0 | COB-ID SYNC | Unsigned32 | rw | 0 |  |
| 1006 | 0 | Cycle period | Unsigned32 | rw | 100000us |  |
| 1007 | 0 | SYNC windows length | Unsigned32 | aw | 500000us |  |
| 1008 | 0 | Device name | String | const | ICAN-4404 | Device name |
| 1009 | 0 | Hardware version | String | const | V1.01 | Hardware version |
| 100A | 0 | Software version | String | const | V1.01 | Software version |
|  |  |  |  |  |  |  |
| 100C | 0 | Guard time | Unsigned16 | rw | 2000ms |  |
| 100D | 0 | Life factor | Unsigned8 | rw | 10 |  |
|  |  |  |  |  |  |  |
| 1010 | 0 | Parameter Store | Unsigned32 | rw |  |  |
| 1011 | 0 | Parameter Restore | Unsigned32 | rw |  |  |
| 1014 | 0 | COB-ID EMCY | Unsigned32 | rw | NodeID+0x  80 |  |
| 1015 | 0 | Inhibit Time EMCY | Unsigned32 |  |  |  |
| 1016 | 0 | Consumer heartbeat  time | Unsigned16 | rw |  |  |
| 1017 | 0 | Producer heartbeat time | Unsigned16 | rw |  |  |
| 1018 | 0 | Identity number | Unsigned8 | ro |  |  |
|  | 1 | Vendor Id | Unsigned32 | ro |  | Vendor code |
|  | 2 | Product Id | Unsigned32 | ro |  | Product Code |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 3 | Revision number | | Unsigned32 | | ro | |  | | Product revision code | |
|  | 4 | Serial number | | Unsigned32 | | ro | |  | | Product serial number | |
|  |  |  | |  | |  | |  | |  | |
|  |  |  | |  | |  | |  | |  | |
|  |  |  | |  | |  | |  | |  | |
| 1200 | 0 | SDO Server Parameter | | Unsigned32 | | ro | |  | |  | |
| 1400 | 0 | RPDO1 Comm  parameter | | Unsigned8 | | ro | | 2 | |  | |
|  | 1 | COB-Id used | | Unsigned32 | | rw | | NodeID+0x  200 | | The RPDO used  COB-ID | |
|  | 2 | transmission type | | Unsigned8 | | rw | | 0xFE | |  | |
| 1600 | 0 | RPDO1 Map parameter | | Unsigned32 | | rw | |  | |  | |
| 1800 | 0 | TPDO1 Comm  parameter | | Unsigned32 | | ro | | 0x05 | |  | |
|  | 1 | COB-ID used | | Unsigned32 | | rw | | NODEID+0 x180 | | TPDO  COB-ID | use |
|  | 2 | transmission type | | Unsigned8 | | rw | | 0xFE | | Transport type |  |
|  | 3 | inhibit time | | Unsigned16 | | rw | | 0 | | PDO transmission time | prohibit |
|  | 4 | event timer | | Unsigned16 | | rw | | 0 | | Time when PDO is transferred | decide |
| 1A00 | 0 | TPDO1 Map parameter | | Unsigned32 | | rw | |  | |  |  |
| 6000 | 0 | Read Inputs 8 Bit | | Unsigned32 | | rw | |  | | Quad switching inputs | |
| 6411 | 0 | Write Analog Output  16 Bit | | Unsigned32 | | rw | |  | | Four analog outputs | |
|  |  |  | |  | |  | |  | |  | |
| Device vendor defines parameters | | | | | |  | |  | |  | |
| 2000 | 0 | Channel Enable | Unsigned8 | | rw |  | 0xf |  |  | Channel enable | |
| 2001 | 0 | Channel Range | Unsigned8 | | rw |  | 0 |  |  | Channel range | |

# Appendix **B: Can open** predefined connection sets (minimum **Can open** devices).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Object | Function code | COB-ID | index | Priority | remark |
| NMT | 0000 | 000H |  | high |  |
| SYNC | 0001 | 080H | 1005H,1006H,1007H |  |  |
| Time Stamp | 0010 | 100H | 1012H,1013H |  |  |
| Emergency | 0001 | 081-0FFH | 1204H,1205H |  | Not realized |
| PDOT1 | 0011 | 181-1FFH | 1800H |  |  |
| PDOR1 | 0100 | 201-27FH | 1400H |  |  |
| PDOT2 | 0101 | 281-2FFH | 1801H |  |  |
| PDOR2 | 0110 | 301-37FH | 1401H |  |  |
| PDOT3 | 0111 | 381-3FFH | 1802H |  |  |
| PDOR3 | 1000 | 401-47FH | 1402H |  |  |
| PDOT4 | 1001 | 481-4FFH | 1803H |  |  |
| PDOR4 | 1010 | 501-57FH | 1403H |  |  |
| SDOT | 1011 | 581-5FFH | 1200H |  |  |
| SDOR | 1100 | 601-67FH | 1280H |  |  |
| Node Guard | 1110 | 701-77FH | 100EH.1016H-1017H | minimum |  |

# Appendix **C:** Specifications

Communication Profile:

1，CA DS301 - Application Layer and Communication Profile;

2，CA DS302 - Framework for Programmable Devices;

CA DS303-1 – Recommendations for Cables And Connectors;

4，CA DS303-2 - SI Units and Prefix Representations; Device Profile

1，DSP401 - I/O Modules;

2，DSP402 - Drives and Motion Control;

3，DSP403 - Human Machine Interfaces(Display And Terminal Devices);

4，DSP404 – Sensors And Data Acquisition Modules;

5，DSP406 – Encoders;

6，DSP408 – Proportional Eventide;