# USBCAN-II C

USB-CAN Converter

# user's manual


Document version：V4.1 （2017/01/09）

# Catalogue

# 1. Function introduction

## 1.1 Functional Overview

USBCAN-II C is an industrial CAN-bus communication interface card integrated with two CAN interfaces. This type of can card can be compatible with the full speed specification of USB2.0 bus. Using USBCAN-II C interface card, PC can be quickly connected to CAN-bus network through USB interface to form a CAN-bus network control node for data processing and data acquisition in the field of CAN-bus network, such as fieldbus laboratory, industrial control, intelligent community, automotive electronic network and so on.USBCAN-II C interface card is a powerful tool for CAN-bus product development and CAN-bus data analysis; at the same time, it has the characteristics of compact size, plug and play, and is also the best choice for users of portable systems. There is a CAN interface electrical isolation protection module on the USBCAN-II C interface card to avoid damage to the equipment due to instantaneous overcurrent / overvoltage and enhance the reliability of the system in harsh environment.。The USBCAN-II C interface card supports 32-bit / 64-bit operating systems such as Windows 2000, XP, 7, 8, 10, and so on. Our company provides users with a unified application programming interface and complete application demonstration code, including VC, VB, Net, Delphi, Labview and C++Builder development routines to facilitate application development.USBCAN-II C interface card can use ECAN Tools, CAN Test and CAN Pro general test software, and can perform the functions of sending and receiving CAN-bus messages and monitoring.

## 1.2 Performance characteristics

- The PC interface conforms to the USB2.0 full speed specification and is compatible with USB1.1
- Integrated 2-way CAN-bus interface, Phoenix terminal connection mode；
- supports CAN2.0A and CAN2.0B frame formats and conforms to the ISO/DIS 11898 specification；
- The baud rate of CAN-bus communication is arbitrarily programmable from 5Kbps to 1Mbps
- Use USB bus power supply, or add external power supply (DC5-24V, 130mA) when the power supply is insufficient;
- CAN-bus interface adopts electrical isolation, isolation module insulation voltage: DC 1500V;
- Maximum received data flow: 8000 fps;
- Support Win2000, WinXP, Win7, Win8, Win10 and other Windows operating systems;

- Support ECANTools, CAN Test, CAN Pro test software;
- Working temperature range: -20℃~+70℃;

## 1.3 Typical application

- CAN-bus network diagnosis and test
- Automotive electronics applications
- Electric power communication network
- Industrial control equipment
- High-speed, large data communication

# 2. Device installation

This chapter introduces the method of connecting the USB-CAN interface card to the computer and the precautions when using the computer to connect the USB-CAN interface card for the first time.

## 2.1 Equipment size

Equipment dimensions: (length, including terminal) 102mm * (width) 63mm * (height) 27mm, the schematic diagram is shown in Figure 2.1


Figure 2.1 Dimensions of USBCAN-II C

## 2.2 Interface definition and function

The USBCAN-II C interface card integrates one USB interface, one DC5-24V auxiliary power interface and two standard CAN-bus interfaces. The CAN-bus interface is led out by a 6 Pin plug-in terminal block, which can be used to connect two CAN-bus network or CAN-bus interface devices.

The location and definition of each interface of USBCAN-II C are shown in Figure 2.2, Figure 2.3, Table 2.1 and Table 2.2.


Figure 2.2 Location of USB and auxiliary power interface

| Pin (From left to right) | Port | Name | Function |
|---|---|---|---|
| 1 | DC 5-24V | DC 5-24V | USBCAN Auxiliary power supply, generally no need to connect |
| 2 | USB | USB | USBCAN power supply,connect to computer |

Table 2.1 Definition of USB and auxiliary power interface


Figure 2.3 Location of CAN-bus interface

| Pin (From left to right) | Port | Name | Function |
|---|---|---|---|
| 1 | CAN1 | L | CAN1_L Signal line（CAN Low） |
| 2 | | P | Shield |
| 3 | | H | CAN1_H Signal line（CAN High） |
| 4 | CAN2 | L | CAN1_L Signal line（CAN Low） |
| 5 | | G | CAN-GND 接地 |
| 6 | | H | CAN1_H 信号线（CAN 高） |

Table 2.2 CAN-bus signal distribution of USBCAN-II C interface card

## 2.3 Driver and software installation

Before installing the driver and software, please ensure that your account for logging in to Windows is a super administrator, or that the user account has the relevant authority to install the driver and software, otherwise the installation may fail.
The method to confirm the permissions of the windows account: control panel-user account.

### 2.3.1 Driver and software installation

Users need to manually install the driver, please enter the "driver" folder in the CD, select the installation file (DriverSetup.exe/DriverSetup64.exe) corresponding to the system (32/64 bit) to manually install.
Users can directly find and install the ECAN Tools, CAN Test and CAN Pro software on the CD. For the specific software installation and usage methods, please refer to the "Software Instructions" in the corresponding folder. Please read "Installation Must Read" before installation.

### 2.3.2 Driver and software uninstall

Users can uninstall the installed device driver by running the above DriverSetup.exe/DriverSetup64.exe and clicking the "Uninstall" button.
Users can uninstall ECAN Tools, CAN Test and CAN Pro software through "Add/Remove Programs" (XP) or "Programs and Features" (win7).

# 3. Use of equipment

## 3.1 Connect with PC

The USBCAN-II C interface card has the characteristics of plug and play, so users can use the USB interface of the PC to directly supply power to the USBCAN-II C interface card; if the USB power supply is insufficient, you need to choose an external power supply.

### 3.1.1 USB bus power supply mode

The USB bus power supply mode is suitable for most applications, for example, when the USBCAN-II C interface card is the only device connected to the USB port.

Connect the PC and the USBCAN-II C interface card directly through the supplied USB cable, and the USB cable provides +5V power to the USBCAN-II C interface card; at this time, the indicator lights PWR and SYS are on, indicating that the device is working properly and is in To be connected.

### 3.1.2 External power supply mode

The external power supply mode is suitable for PCs that use a USB bus hub (HUB), or have connected multiple USB terminal devices, resulting in a USB port that cannot provide sufficient current to the USBCAN-II C interface card.

Use an external power supply (DC+5V, 200mA) to connect to the DC5V power socket of the USBCAN-II C interface card. At this time, the indicators PWR and SYS are on; then connect the PC and the USBCAN-II C interface card through the supplied USB cable , USBCAN-II C interface card can work normally.

## 3.2 Connect with CAN-bus

When USBCAN-II C is connected to the CAN bus, just connect CAN_H to CAN_H and CAN_L to CAN_L to establish communication.The CAN-bus network adopts a straight-line topology, and the two furthest terminals of the bus need to install 120Ω terminal resistance; if the number of nodes is greater than 2, the intermediate nodes do not need to install 120Ω terminal resistance. For branch connections, the length should not exceed 3 meters. The connection of CAN-bus is shown in Figure 3.1.

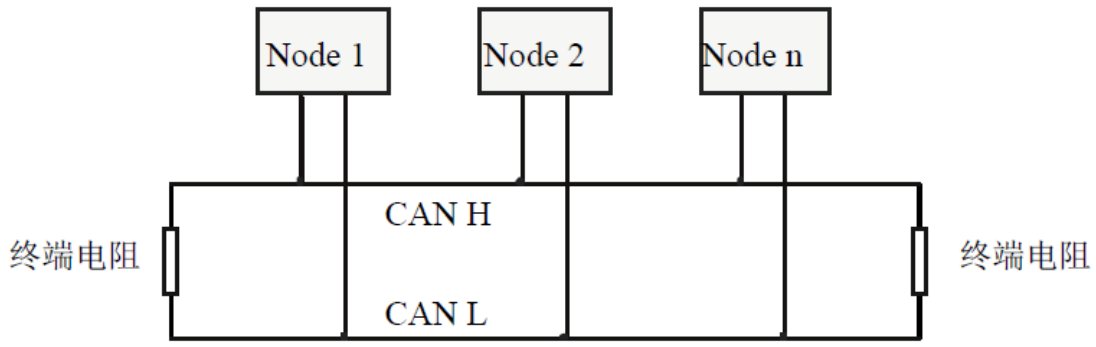



Figure 3.1 Topology of CAN-bus network

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

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

Table 3.1 Baud rate and maximum bus length reference

## 3.3 CAN bus termination resistance

In order to enhance the reliability of CAN communication and eliminate CAN bus terminal signal reflection interference, the two farthest endpoints of the CAN bus network usually need to add terminal matching resistors, as shown in Figure 3.2. The value of the termination matching resistance is determined by the characteristic impedance of the transmission cable. For example, the characteristic impedance of the twisted pair is 120Ω , then the two endpoints on the bus should also integrate 120Ω termination resistors. The USBCAN-II C interface card uses 82C251 transceiver. If other nodes on the network use different transceivers, the terminal resistance must be calculated separately.

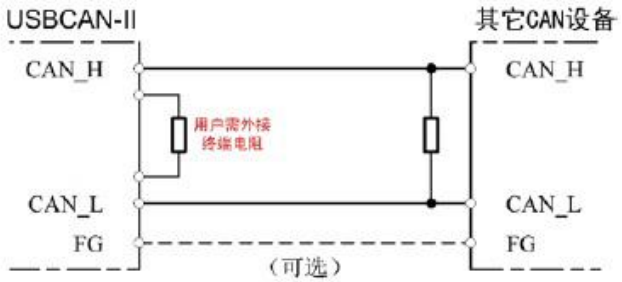



Figure 3.2 USBCAN-II C connected with other CAN node devices

Note: USBCAN-II C has integrated 120Ω terminal resistance, you can choose whether to connect the resistance to the bus through the DIP switch, the DIP switch is next to the 6pin terminal, R1 and R2 are the terminal resistance of CAN1 and CAN2, respectively. Position to activate the resistor.

## 3.4 System status indicator

The USBCAN-II C interface card has a PWR indicator, a SYS indicator, a CAN1 indicator, and a CAN2 indicator to indicate the operating status of the device. The specific indication functions of these 4 indicators are shown in Table 3.2. When these 4 indicators are in various states, the state of the CAN bus is shown in Table 3.3.

| Indicator light | Color | Indication status |
|---|---|---|
| PWR | Green | Power indicator |
| SYS | Green | System indicator |
| CAN1 | Green | CAN1 signal indication |
| CAN2 | Green | CAN2 signal indication |

Table 3.2 Indicators of USBCAN-II C interface card

After the USBCAN-II C interface card is powered on, PWR is lit, and the system initialization status indicator SYS is lit, indicating that the device has been powered and the system is initializing; otherwise, it indicates that there is a system power failure or a serious error has occurred in the system.

After the USB interface is connected normally, when the USB bus is transmitting data, the USB signal indicator SYS will flash.

When CAN1 or CAN2 send and receive data, the corresponding CAN1 and CAN2 indicator lights will flash.

| Indicator light | Status | Indication status |
|---|---|---|
| PWR | Bright | Power supply is normal |
| | Not bright | Power supply is abnormal |
| SYS | Always bright | Device initialized, standby |
| | Not bright | Device initialization failed |
| | Flicker | USB interface data transmission |
| CAN1、CAN2 | Not bright | No data transmission on CAN channel |
| | Flicker | Corresponding CAN channel has data transmission |

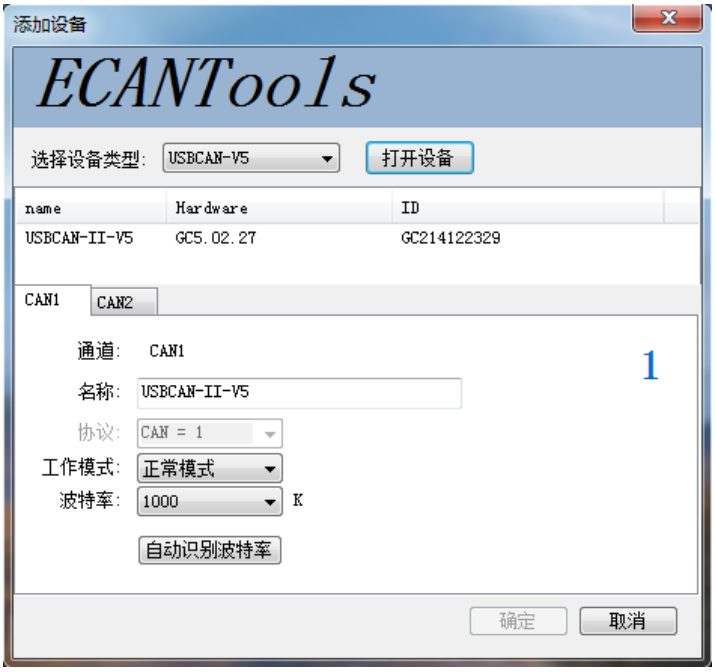
Table 3.3 Indicator status of USBCAN-II C interface card

# 4. ECAN Tool software use

ECAN Tools software is a special debugging analysis software developed by our company for the windows platform. With this software, users can intuitively and quickly send and receive CAN bus data. The software is extremely easy to use and has extensive extension functions, so users can get started quickly. This chapter will introduce the main functions of ECAN Tools。

## 4.1 Software launch

1. If the user has installed ECAN Tools software, you can open the software by double-clicking the "ECAN Tools" icon on the desktop.



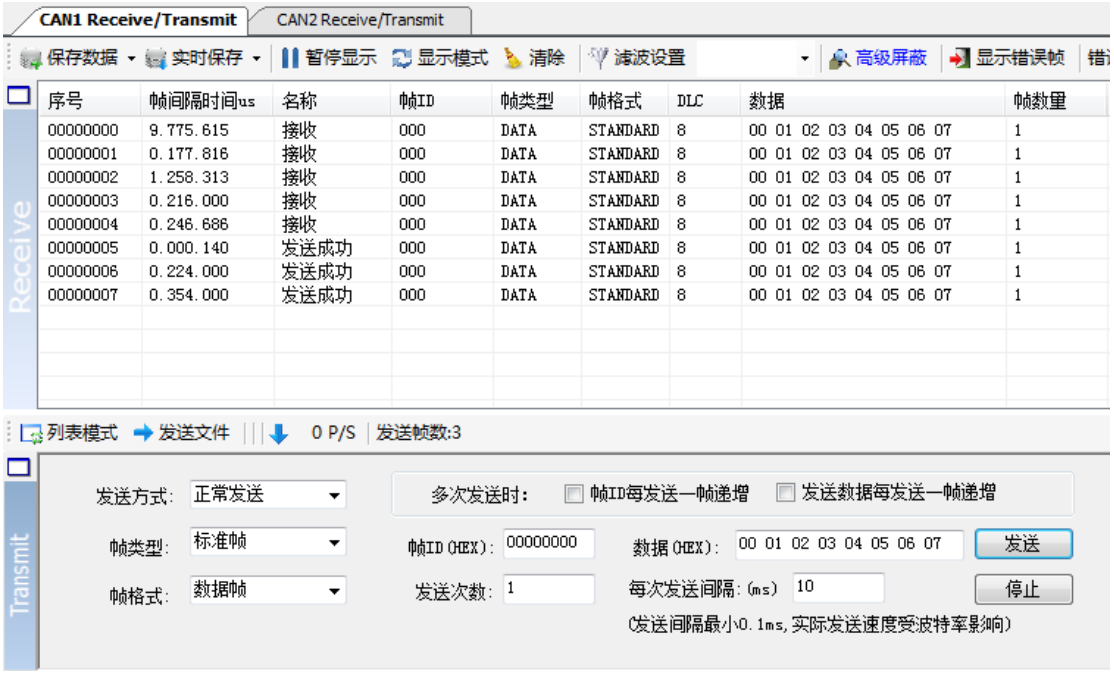

2. After selecting the corresponding device type, click "Open Device" and the CAN device that has been plugged into the USB port of the computer will appear in the device list.

3. Select the working mode. The software provides 3 working modes to choose from: normal mode, listening only mode, spontaneous self-receiving mode. The normal mode is used to send and receive data using the software normally; the listen-only mode is used to monitor the bus and cannot send data. The USBCAN device does not serve as a CAN bus node and does not send replies and clock signals; the spontaneous self-receive mode is used to test whether the CANCAN communication of the USBCAN device is normal. .

4. Select the baud rate. The user needs to set according to the baud rate of the bus actually connected. If the baud rate does not match the bus, it cannot communicate normally or even interfere with the bus. If you need to customize the baud rate, please contact our company. When the user does not know the bus baud rate, the automatic baud rate recognition function can also be used for scanning.

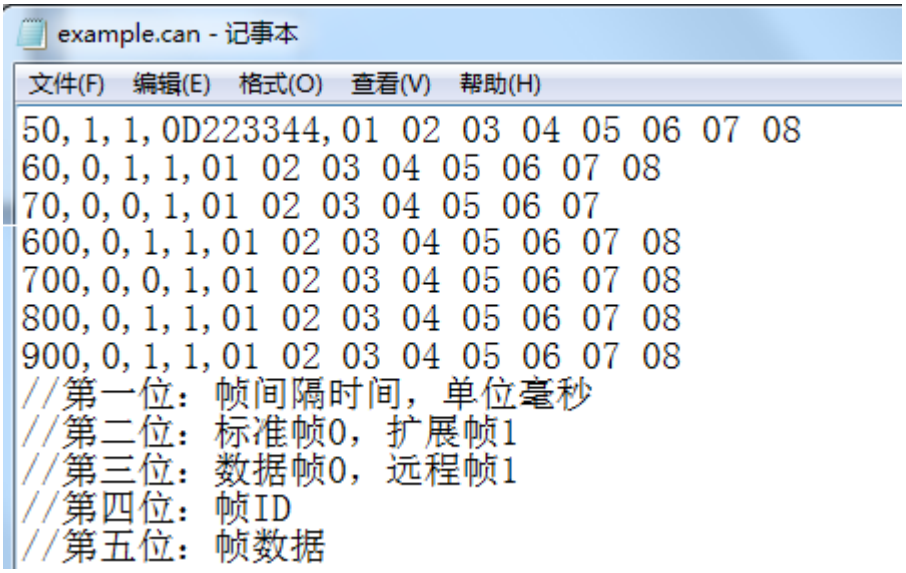
## 4.2 Data sending and receiving

The receiving and sending interface is the main function of ECAN Tools. In this interface, the user can intuitively see the received CAN message and send the data to the bus by ordinary sending or list sending.



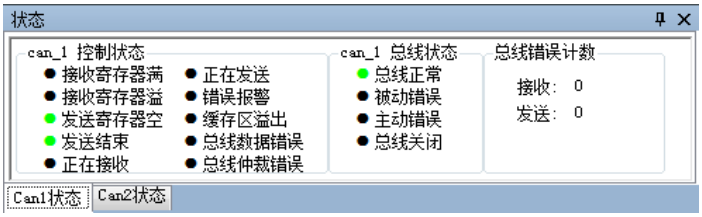

In addition to the above basic functions, the software also has the function of sending files. File sending is divided into ordinary file sending and batch file sending. Common file sending can send files to the nodes on the bus, users can develop CAN bus program burning software based on this function; batch file sending can send the batch file (.can) saved by the user according to the time sequence and data at the time of saving Return to the CAN bus. For details about saving batch files, see 4.4.

**Examples of batch file sending applications:**



## 4.3 Bus analysis function

With bus diagnosis function, it can detect bus error frames and bus arbitration lost bits, and realize a more comprehensive CAN bus analysis.




**CAN bus status display:** indicates that the current CAN bus status includes: bus normal, passive error, active error, bus hang.

**Internal FIFO of the CAN controller overflows:** bus messages are too dense for a certain period of time, resulting in USBCAN too late to receive and message loss.

CAN controller error alarm: multiple transmission errors or reception errors occur on the bus, the error counter exceeds the alarm threshold, and the error count value can be displayed.

**CAN controller negative error:** Multiple transmission errors or reception errors occur on the bus, causing the CAN controller to enter an error negative state, and can display the error count value.

**CAN controller arbitration lost:** The node and other devices failed in the competition for the bus. The pending message will try to be sent when the next bus is idle, and the corresponding error bit information can be captured and displayed.

**CAN controller bus error:** once a transmission error or a reception error occurs on the bus, the value of the error counter will be accumulated and can capture the wrong bit information, such as ACK, CRC, bit definition error information.

**CAN data buffer overflow:** Since the data has not been exchanged with the PC for a period of time, the number of messages stored by the device exceeds the hardware capacity limit, and the newly received message will overwrite the oldest message.

## 4.4 Other functions

ECAN Tools software is extremely rich in extension functions, and they are very convenient and easy to use. Flexible use of these extension functions can help users analyze data conveniently and quickly.


**Save data / real-time save:** can save the receiving list, save format: txt, can, excel, binary.
**Pause display:** You can pause the scrolling of the reception list.
**Display mode:** Scroll mode and list mode are available. List mode can classify data in real time according to rules.
**Clear:** You can clear the data in the current receiving list.
**Filter setting:** Multi-segment filtering is realized by editing the filter ID.
**Advanced blocking:** By blocking the ID, the blocked ID is not displayed.
**Display error frames:** You can show/hide error frames on the bus.
**Please note:** For more detailed functions of the ECAN Tools software, please see "ECAN Tools Software Instructions"

# 5. CAN Test software use

CAN Test software is a debugging analysis software for windows platform. With this software, users can intuitively and quickly send and receive CAN bus data. The software is extremely easy to use and has extensive extension functions, so users can get started quickly.

## 5.1 Software launch

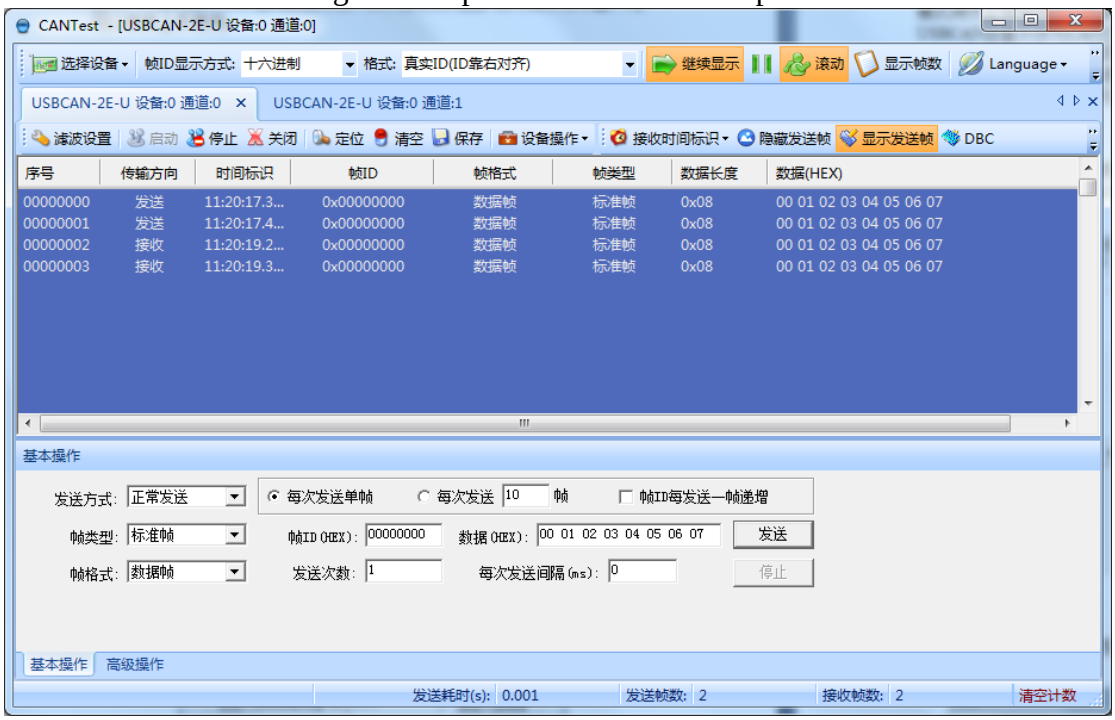
1. If the user has installed the CAN Test software, you can open the software by double-clicking the "CAN Test" icon on the desktop.


2. Click to select the device, after selecting the USBCAN-2E-U device type, you can enter the parameter setting page。

3. Select the device index number and the number of CAN. When users insert multiple USBCAN of the same model on the same computer, they need to select the corresponding index number to distinguish which device to open. Select the number of CAN channels to choose to open the device CAN1 or CAN2, if you want to open at the same time, you need to check "select all CAN".
4. Select the working mode. The software provides 2 working modes: normal mode and listening only mode. The normal mode is used to send and receive data using software normally; the listen-only mode is used to monitor the bus and cannot send data. The USBCAN device does not act as a CAN bus node and does not send replies and clock signals.
5. Select the baud rate. The user needs to set according to the baud rate of the bus actually connected. If the baud rate does not match the bus, it cannot communicate normally or even interfere with the bus.
6. Click OK and start to enter the software.

## 5.2 Data sending and receiving

Receiving and sending CAN bus data is the main function of CAN Test. On the main interface of the software, users can intuitively see the received CAN message and send data to the bus through basic operation or advanced operation.

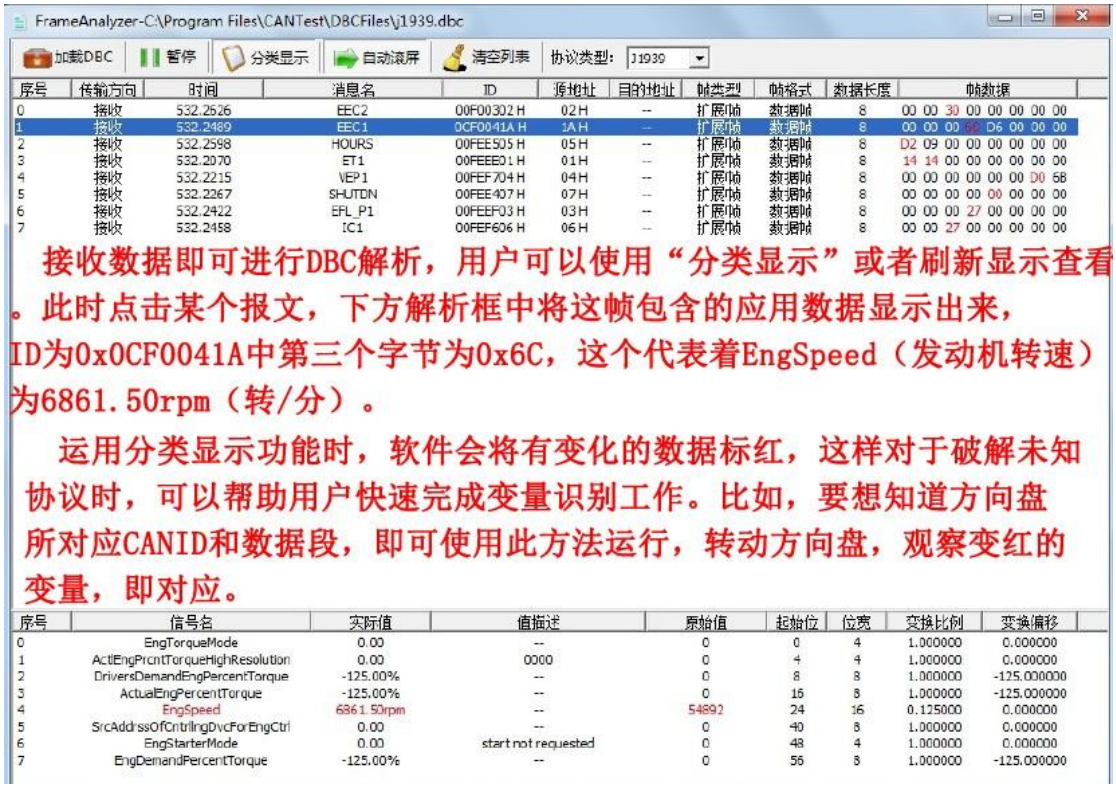


## 5.3 DBC file parsing function

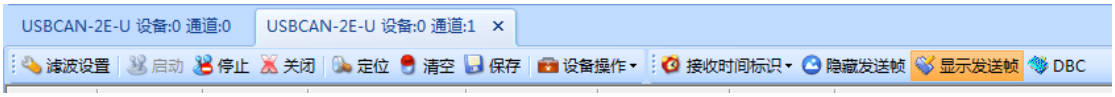
The core function of CAN Test software-DBC file analysis. The received bus data can be analyzed in detail by loading DBC files. The software comes with DBC files parsed by the CANopen protocol, J1939 protocol and BMS protocol. Users can load these files to match the data of the corresponding protocol standards. The detailed meaning is resolved. This function is especially suitable for users of automotive electronics. Using the J1939 protocol DBC analysis can parse the car data that meets the J1939 protocol into specific meanings, such as vehicle speed, speed and other specific information.

If users have their own DBC files, CAN Test software can also be used to load and parse the data in accordance with the protocol. CAN Test software can load Motorola and Intel format DBC files.

CAN Test software does not provide DBC file editing function.



## 5.4 Other functions




Filter setting: temporarily unavailable.

Start/stop/close: can operate the current equipment.

Positioning: You can quickly locate the first line of the receiving list.

Clear: You can clear the data in the current receiving list.

Save: You can save the data in the current send/receive list.

Device operation: You can see the hardware information of the current device.

Receiving time indicator: select whether to display the original value or the actual time.

Hide/Show Send Frame: The sent data can be displayed or not displayed in the receiving list.

DBC: As introduced in 5.3.

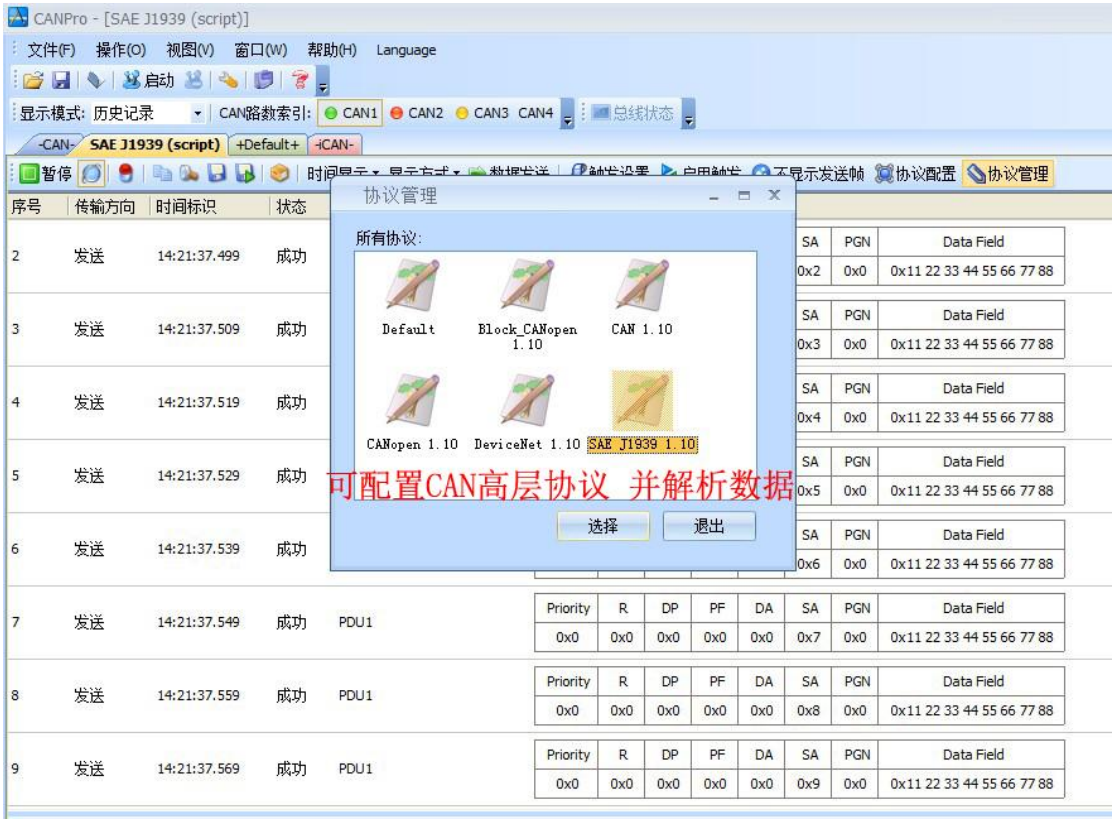
**Please note**: For more detailed functions of the CAN Test software, please see "CAN Test Software Instructions"

# 6. CAN Pro software use

The core function of CAN Pro software is similar to CAN Test software except for the basic sending and receiving functions. CAN high-level protocol analysis.

The protocol analysis function of the CAN Pro software supports three CAN bus high-level protocols: CANopen, SAE J1939, and Devicenet.

The protocol parsing function can load the corresponding standard protocol plug-in. When the data on the bus conforms to the loaded protocol format, the software will automatically dismantle and display this piece of data according to the protocol. This feature can help users who do not understand the protocol to quickly learn and speed up the project process.




**Please note**: For more detailed functions of CAN Pro software, please see "CAN Pro Software Instructions.

# 7. Secondary development

Our company provides standard interface function libraries for secondary development users, including: ECANVCI.h, ECANVCI.lib, ECANVCI.dll. The interface function library is in a standard format, and users can declare and call these interface functions in VC, VB and other programming environments. For specific usage methods, please refer to the "ECAN Dynamic Library User Manual". Figure 8.1 shows the names of commonly used structures and the call flow of the function library.

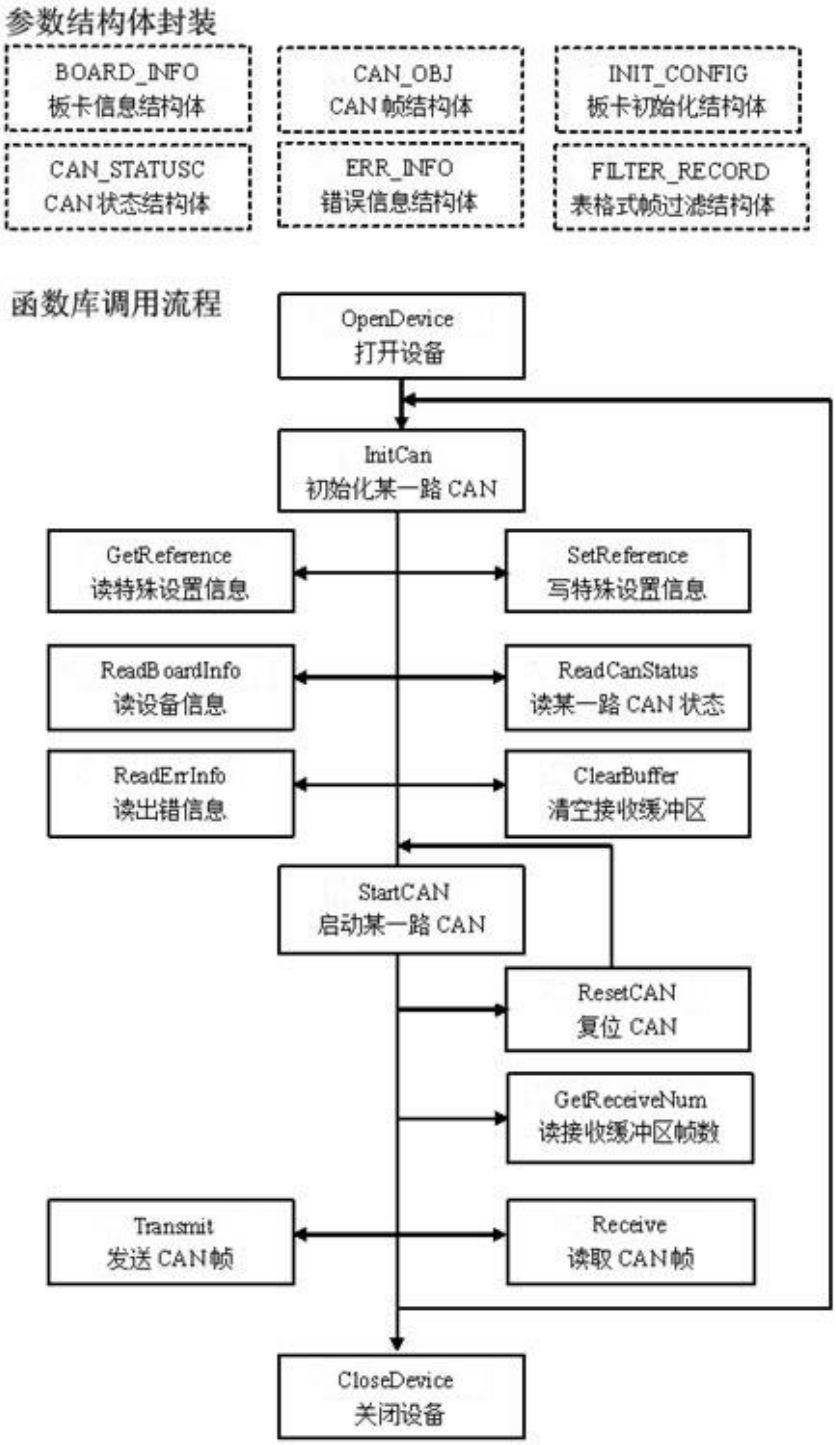



Figure 8.1 Second development function call

# 8. Technical specifications

| | |
|---|---|
| **Connection method** | |
| PC side | USB port，Types of A |
| CAN side | OPEN6 Phoenix Terminal |
| **Interface characteristics** | |
| USB interface | USB2.0 full speed interface, compatible with USB1.1, USB3.0 |
| CAN interface | Follow ISO 11898 standard, support CAN2.0A/B |
| CAN Baud rate | 5Kbit/s~1Mbit/s |
| Electrical isolation | 1500V，DC-DC |
| CAN Terminating resistor | Has been integrated, choose whether to enable through the dial switch |
| **Power supply** | |
| Supply voltage | +5V DC （USB interface） |
| Supply current | Max 130mA |
| **Environmental test** | |
| Operating temperature | -40℃~+85℃ |
| Working humidity | 15%~90%RH， No condensation |
| Protection level | IP 20 |
| **Basic Information** | |
| Dimensions | 95mm *65mm *26mm |
| Weight | 100g |

# 9. Common problem

**1. When opening CAN Test or CAN Pro, it prompts "Failed to open device"?**
The possible causes of such failures are: controlcan.dll is not replaced or the USBCAN device driver is not installed normally. Check the USBCAN device properties in the device manager of the PC to see if "!" or "?" is in front of the USBCAN device; if so, check the hardware/software conflict and reinstall the USBCAN device driver.

**2. Is it necessary to use 120Ω terminal matching resistor?**
It is recommended that the 120Ω terminal matching resistor be used to absorb the end reflection and provide a stable physical link. When the single-node spontaneous self-receiving test is performed, the 120Ω terminal resistance must be connected to form a loop, otherwise the spontaneous self-receiving test cannot be performed. The USBCAN-II C interface card has been connected with a 120Ω termination resistor.

**3. Can a computer install multiple USBCAN interface cards?**
The old version of the interface does not support simultaneous operation of multiple cards, but the current USBCAN interface card supports the simultaneous operation of multiple PC-CAN interface cards of the same model.

**4. What is the highest data conversion rate of the USBCAN-II C interface card?**
The single CAN channel of the USBCAN interface card supports up to 8000 fps CAN bus data conversion. The frame mentioned here refers to a standard frame of 8 data frames. If it is less than 8 bytes of data or a remote frame may be faster. In addition, the maximum data traffic will be limited by PC performance.

**5. Why does the CAN status indicator not light?**
Because all operations of the USBCAN interface card are controlled by the PC, the CAN status indicator will be meaningful only after the PC sends the command to start CAN communication.

**6. Why does the system operate illegally when calling interface functions?**
First of all, please read the function description carefully when using interface functions to ensure that the input parameters are legal, pay special attention to the transfer of pointers (addresses), or refer to the provided sample programs. If the problem still cannot be solved, please contact our technical support.

**7. How to set the communication baud rate of PC-CAN interface card?**
Provide a set of commonly used baud rate settings. If you want to use other baud rates, please refer to "CAN-bus General Test Software and Interface Function Library User Manual" to calculate by yourself. Note: The CAN controller of the USBCAN interface card uses a 24MHz clock, and the user needs to calculate the baud rate according to the clock frequency.

**8. Does the system enter the standby or sleep state affect the reception?**
Will have an impact. At this time, all processing will stop, which may cause a hardware receive buffer overflow error. If there is a program to open the device, it will try to prevent the system from entering standby or sleep state, so as to ensure that the system works normally. When using a USBCAN interface card, please disable the

system's standby and sleep functions.

**9. How to deal with errors in the application?**

The errors are mainly divided into two types: function call errors and CAN-bus communication errors. Function call errors are generally caused by parameter errors, such as: device number out of range, type number error, etc. The error number returned by Win32 function GetLastError is 87, and there are operations on unopened devices, which is actually an illegal handle operation. According to the specific function call situation, there are corresponding Win32 standard error codes provided. The user can use GetLastError to perform error analysis. This part of the debugging work should generally be completed at design time.

For CAN-bus communication errors, generally caused by the CAN network, it may also be caused by improper user settings, such as: inconsistent baud rate settings, call the send function without starting the CAN controller, etc. Most of the errors have already been dealt with in the device driver. If you want to perform deeper error analysis and processing, you can call the ReadCANStatus function.

Another thing to note is the data overflow interrupt error, which has two possibilities: (1) Software receive buffer overflow. This shows that the application cannot process the received data in a timely manner. At this time, the user should optimize the application or change the communication strategy. (2) The hardware receive buffer overflows. This error is caused by the PC interrupt delay at the receiving end being too large. It can only be solved by improving the performance of the computer or coordinating the remaining nodes to appropriately reduce the sending speed.

**10. What should be paid attention to when opening and closing the device?**

The USBCAN-II C interface card provides 2 CAN ports, and users can operate different ports in the same program. The USBCAN-II C interface card does not allow sharing devices to be opened. The same device cannot be opened multiple times by different processes by calling the OpenDevice function. The OpenDevice and CloseDevice functions generally only need to be called once during application initialization and exit. When closing the device, if the current port can no longer be used, you should first call the ResetCAN function to get the current port out of the CAN bus. The device driver will only automatically call ResetCAN to exit the CAN bus connection when the last device handle is closed.

**11. How to use the interrupt mode to operate the communication card?**

The USBCAN interface card does not provide an interface for directly operating the interrupt, because the interrupt is already handled in the driver. Most of the reasons for the need to interrupt the operation in the application program: the program does not know when the data can reach the device, and needs to get a trigger to receive the message to read the data from the buffer. The general solution to this problem is to use multithreading (or multitasking). That is, a new thread is started, and the Receive function is cyclically called in the thread to query the receive buffer. The blocking mechanism has been implemented inside Receive. When there is no data in the buffer, the calling thread will be suspended. At this time, the CPU time will not be occupied, and the application can still handle other transactions.

**12. How to use Transmit send function better?**

The driver of the USBCAN interface card provides about 128 frames of transmit buffer FIFO, and each Transmit call sends up to about 128 frames of data. The sending speed of the sending device is determined by the performance of the current computer software and hardware. Generally, the continuous sending speed is about 2000 fps (standard data frame 11Bytes, 1Mbps). If the sending speed is too fast, it may cause the data of the remote receiving device to overflow and lose response. The user can add a delay in the application programming to reduce the transmission speed. Each frame has a timeout limit during transmission. The timeout period is about 2 seconds when a single frame is sent. The timeout for the last frame when sending multiple frames at a time is 2 seconds, and the rest is 1 second. The transmission timeout usually occurs when the CAN bus is busy and the current node priority is low. It is not a function call or communication error. The user can program the retransmission (generally, transmission timeout events rarely occur in low- and medium-speed networks). Therefore, pay attention to ensure that the CAN bus occupancy should not exceed 60-70% of the bus capacity during system design.

**13. How to use Receive function better?**

The device driver provides 100,000 frames of software receive buffer, which provides application programmers with sufficient response processing time. When the software receives buffer data overflow, the device driver will call ResetCAN to reset the CAN bus, and at the same time set the CAN status data overflow interrupt flag bit. Note that both software buffer overflow and CAN controller hardware buffer overflow use this flag bit.

The Wait function provides Wait parameters suitable for multi-threaded programming. The function encapsulates a blocking function. The Wait parameter has the same meaning as the Win32 WaitForSingleObject dwMilliseconds parameter (please refer to the Win32 API description). It specifies a timeout return time for Receive in milliseconds.

When Wait is 0, the function returns the number of successfully read frames immediately when the function is called, or 0 if the receive buffer is empty. When Wait is non-zero, if there is data in the receive buffer when the function is called, the number of successfully read frames will be returned immediately. If the receive buffer is empty at this time, the function will wait for a specified timeout to arrive or receive data before returning successful reception. Number of frames. When Wait is 0xFFFFFFFF, it is waiting indefinitely until data is received. It is recommended not to set Wait too large. Infinite waiting should pay more attention.

When nFrames is equal to 0, the function actually returns a notification message. It does not require reading the receive buffer. It is a special skill. Note: If the Receive function is called in the main thread and Wait is not 0, it may cause the application to temporarily lose its response. If it is received by query, Wait should generally be set to 0.

# Appendix CAN2.0B protocol frame format

## **CAN2.0B** Standard frame

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

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

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

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

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

## **CAN2.0B** Extended Frame

CAN extended frame information is 13 bytes, including two parts, information and data part. The first 5 bytes are the information part。

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| 字节 1 | FF | RTR | × | × | DLC（数据长度） | | | |
| 字节 2 | （报文识别码） ID.28－ID.21 | | | | | | | |
| 字节 3 | ID.20—ID.13 | | | | | | | |
| 字节 4 | ID.12—ID.5 | | | | | | | |
| 字节 5 | ID.4－ID.0 | | | | | × | × | × |
| 字节 6 | 数据 1 | | | | | | | |
| 字节 7 | 数据 2 | | | | | | | |
| 字节 8 | 数据 3 | | | | | | | |
| 字节 9 | 数据 4 | | | | | | | |
| 字节 10 | 数据 5 | | | | | | | |
| 字节 11 | 数据 6 | | | | | | | |
| 字节 12 | 数据 7 | | | | | | | |
| 字节 13 | 数据 8 | | | | | | | |

Byte 1 is frame information. The 7th bit (FF) indicates the frame format. In the extended frame, FF=1; the 6th bit (RTR) indicates the type of the frame, RTR=0 indicates the data frame, RTR=1 indicates the remote frame; DLC indicates the data The actual data length when framing.
Bytes 2 to 5 are message identification codes, and the upper 29 bits are valid.
Bytes 6 to 13 are the actual data of the data frame, the remote frame is invalid.