USB-CAN-FD

From Waveshare Wiki
Jump to: navigation, search

Introduction

The USB-CAN-FD is an industrial-grade high-performance USB to CAN FD adapter, CAN/CAN-FD bus communication interface card, and CAN/CAN-FD protocol data analyzer. Onboard dual independent CAN FD interfaces with electrical isolation and multiple protection circuits. Supports Windows XP/7/8/10/11 system, comes with drivers, CAN FD Tools related software, secondary development examples, and tutorials.
It can be connected to the PC or industrial control host via a USB port to realize transceiver control, data analysis, collection, and monitoring of the CAN/CAN FD bus network. It is compact in size and easy to use, which can be used for learning and debugging of CAN/CAN FD bus, as well as for secondary development and integration into

USB-CAN-FD



(https://www.waveshare.com/usb-can-fd.htm)

USB To CAN/CAN FD, CAN/CAN FD

USB-CAN-FD-B



(https://www.waveshare.com/usb-can-fd.htm?sku=28745)

USB To CAN/CAN FD, CAN/CAN FD

various industrial, power communication, and intelligent control applications that require CAN/CAN FD bus communication.

Specifications

Model		USB-CAN-FD-B (https://www.waveshare.com/usb-can-fd.htm?sku=28745)	USB-CAN-FD (https://www.waveshare.com/usb-can-fd.htm)
Product Type	Industrial Grade	Industrial Grade: USB to CAN FD interface converter, CAN/CAN FD bus communication card, CAN/CAN FD protocol data analyzer	
USB Port	Working Voltage	5V (Directly powered via USB port, no additional power supply required)	
	Interface Form	USB-B Port	
CAN/CAN FD Interface	CAN/CAN FD Channels	Dual channels: CAN1 and CAN2 (Independent of each other, do not affect each other, full isolation, isolation voltage 3000V DC)	
	Interface Form	Terminal interface (OPEN6/KF2EDG-6P, 5.08mm pitch)	
	Matching Resistor	Each CAN/CAN FD channel has two built-in 120Ω terminal resistors, which can be enabled by switch	
	Baudrate	100Kbps~5Mbps	
	Protocol Support	CAN2.0A and 2.0B, ISO 11898-1 CAN FD Protocol V.1.0	
	Transfer Speed	The receiving and sending speed of each CAN/CAN FD channel can reach 20000 frames/s and 5000 frames/s	
	Transmit Buffer	2000 frames receiving buffer and 1000 frames sending buffer per channel (automatically retransmit when the transmission fails)	1500 frames receiving buffer and 64 frames sending buffer per channel (automatically retransmit when the transmission fails)
Indicator	PWR	Power Indicator	
	SYS	System status indicator, normally off; keeps on when there is an error	
	CAN1	CAN1 channel indicator (blinking when sending and receiving data)	
	CAN2	CAN2 channel indicator (blinking when sending and receiving data)	
System Support	Windows	Windows XP/7/8/10/11 (32-bit and 64-bit)	
	Linux	Raspberry Pi OS, Ubuntu (Jetson Nano), VMware virtual machines, etc.	Does not support Linux
Operating Temperature		-40 ~ 85℃	
Case Material		Aluminum alloy case + 3D flame-retardant insulating sheets on both sides provide better protection against metal tip discharge, improve product safety, and extend service life	
Dimensions		104 × 70 × 25 (Unit: mm)	

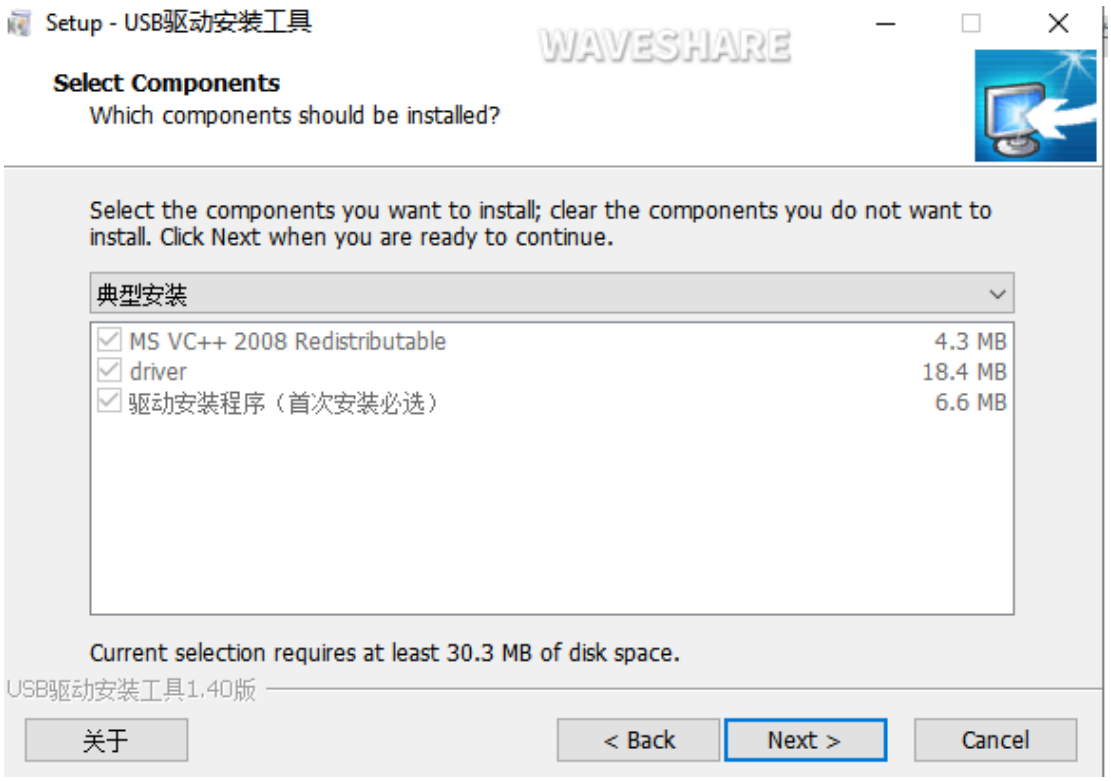
Windows Tutorial

Both USB-CAN-FD and USB-CAN-FD-B support use on Windows systems, and the tutorial on Windows systems is shared.

Install Driver

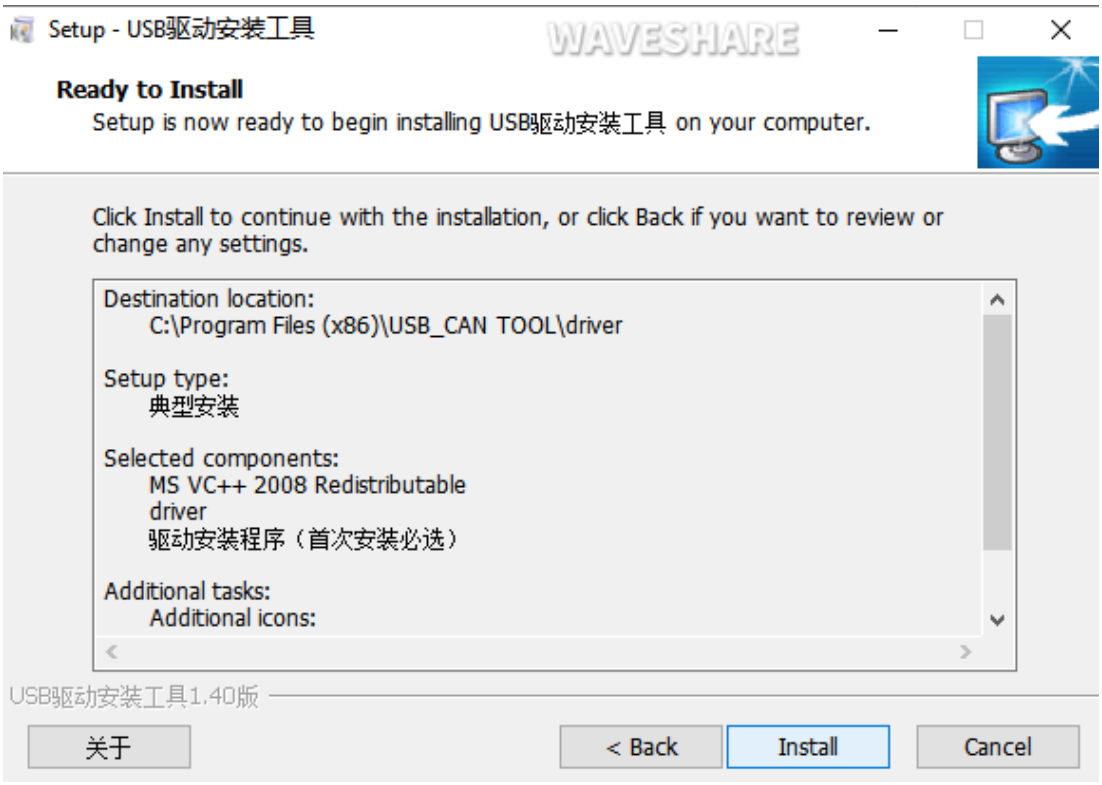
Use the USB cable to properly connect the USB-CAN-FD Smart Interface Module to the PC. Once the hardware is detected, Windows will automatically run the installation wizard called "Discover New Hardware". Download Windows Driver (https://files.waveshare.com/wiki/USB-CAN-FD/Driver/EN-USB-CAN-FD-driver-1.40.exe.zip).

- Click Next to continue



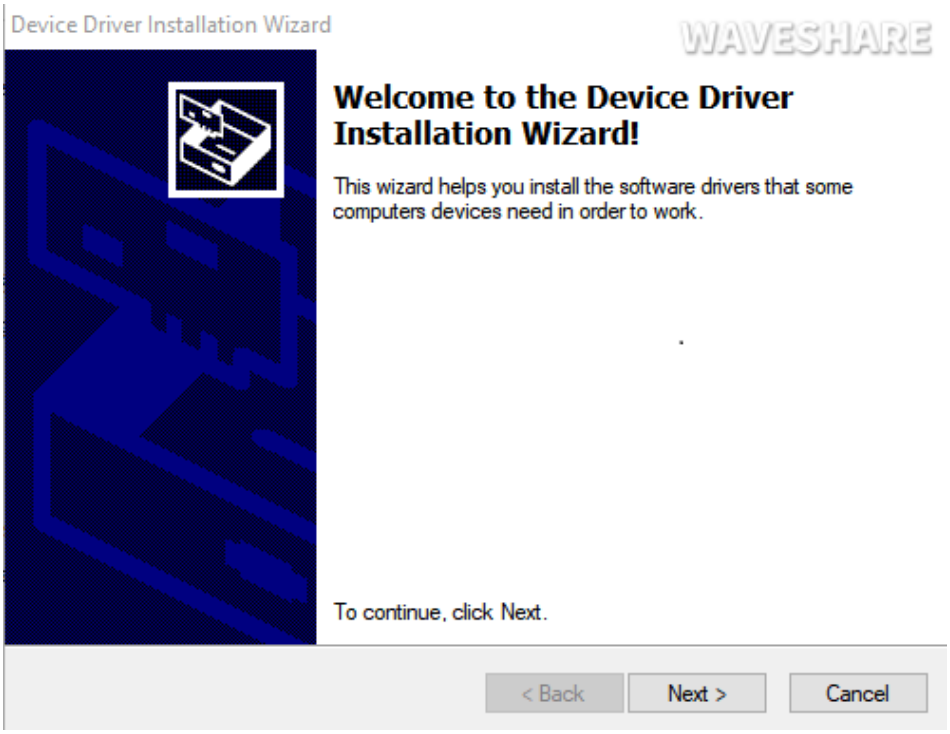
(/wiki/File:Image4-USB-CAN-B-Driver_(1).png)

- Click Install



(/wiki/File:Image4-USB-CAN-B-Driver_(2).png)

- Click Next



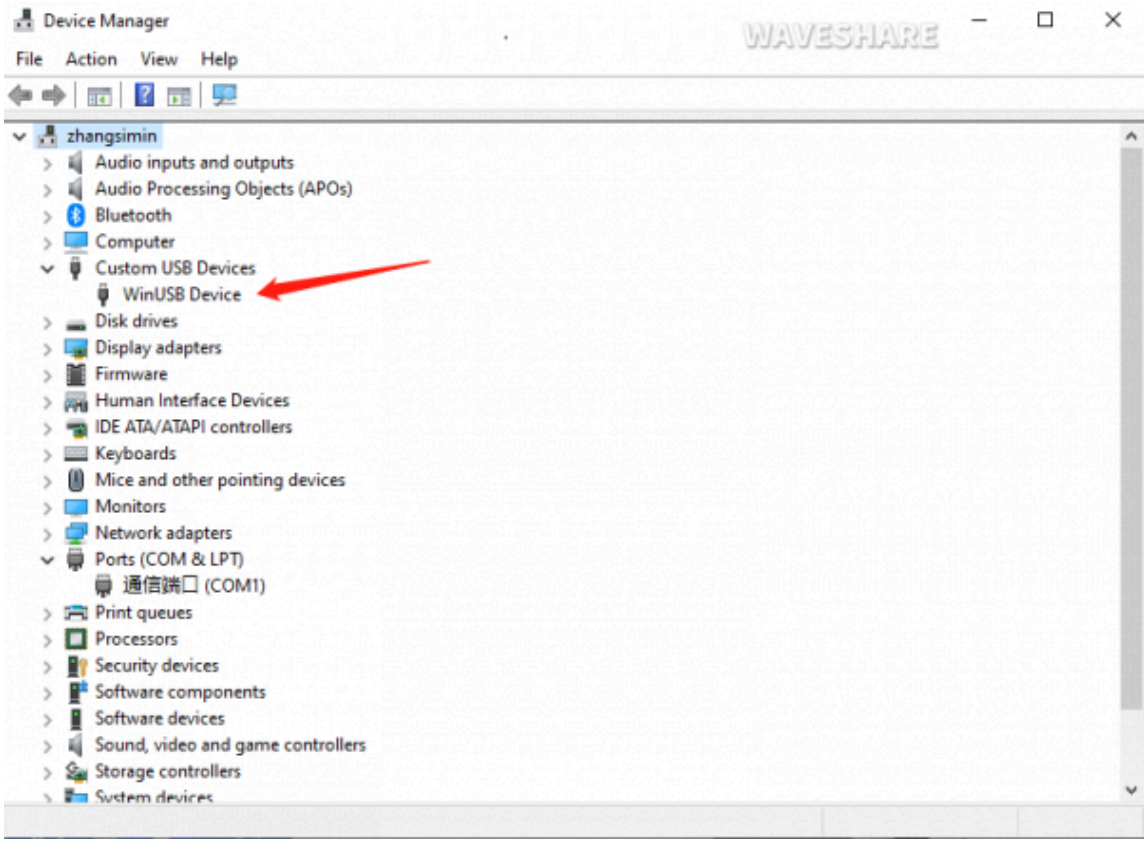
(/wiki/File:Image4-USB-CAN-B-Driver_(3).png)

- Complete installation



(/wiki/File:Image4-USB-CAN-B-Driver_(4).png)

- After successful installation, you will see the WinUSB Device in the Device Manager

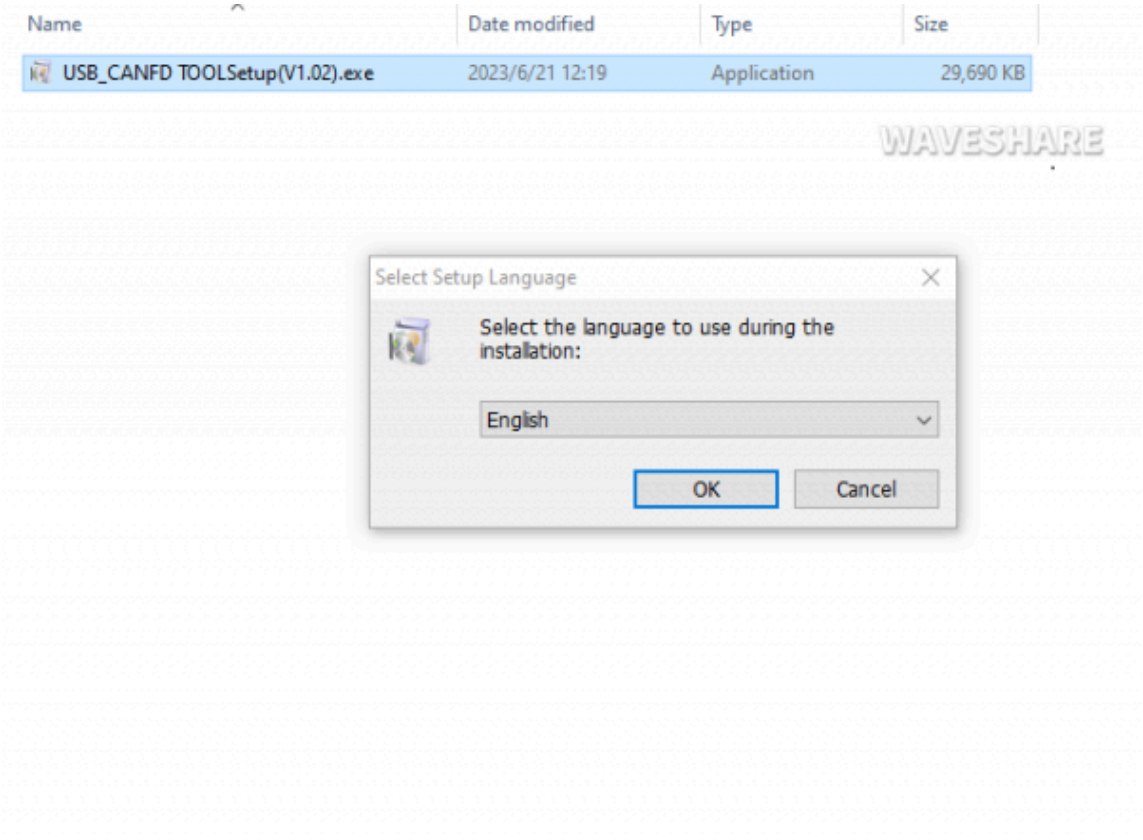


(/wiki/File:USB-CNA-B-driver.png)

USB-CAN-FD Tool Software (https://files.waveshare.com/wiki/USB-CAN-FD/tool/USB_CANFD%20TOOLSetup(V1.17).exe.zip)

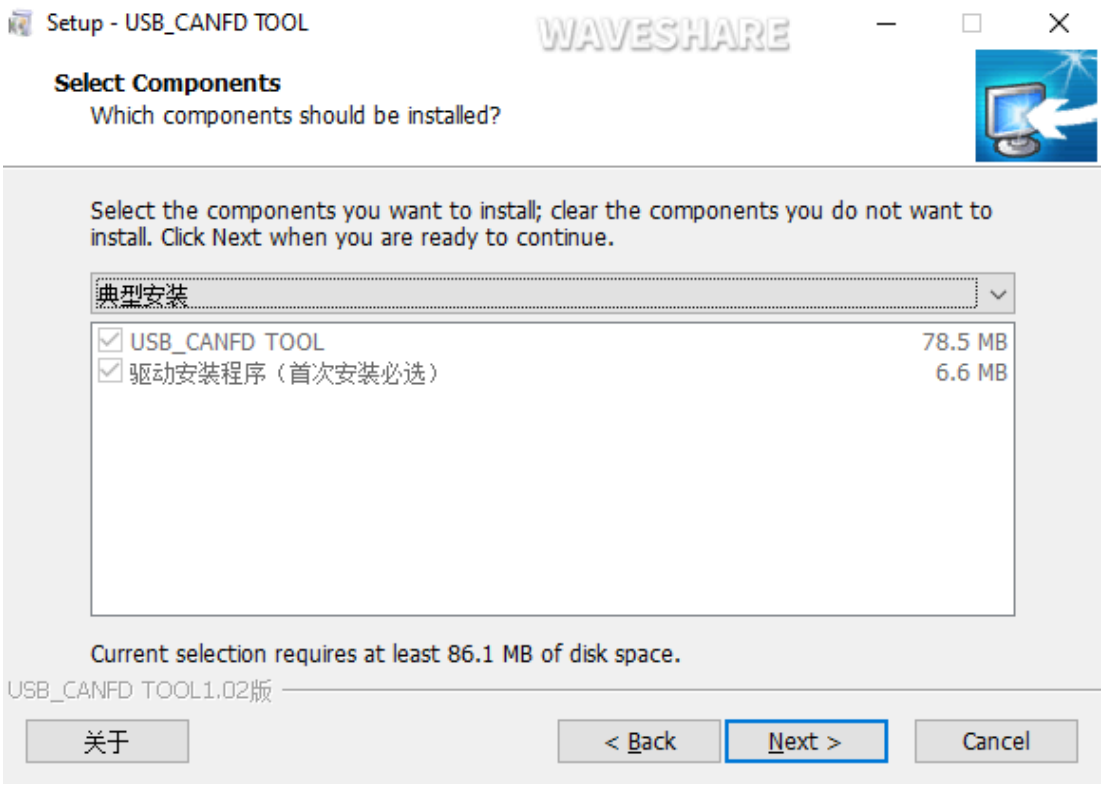
Download & Install (https://files.waveshare.com/wiki/USB-CAN-FD/tool/USB_CANFD%20TOOLSetup(V1.17).exe.zip)

- Choose the language according to your habits, here is an example of English



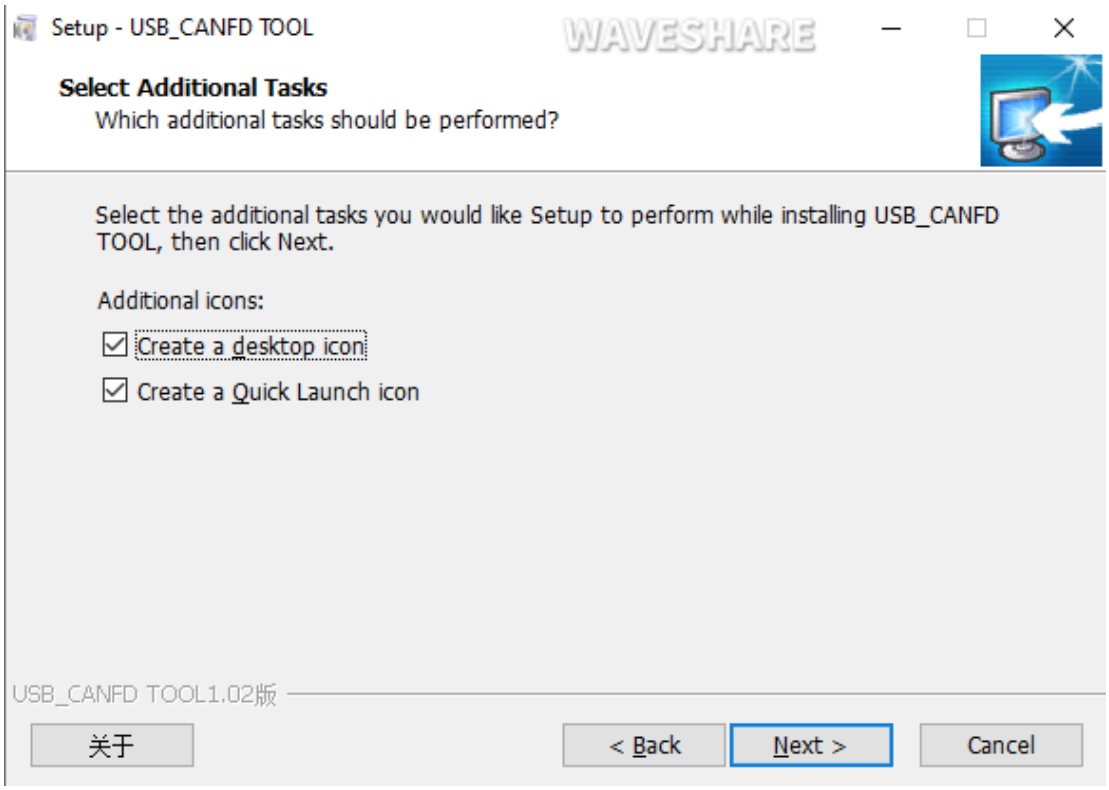
(/wiki/File:USB-CAN-FD_Tool01.png)

- Select the default option, and click Next



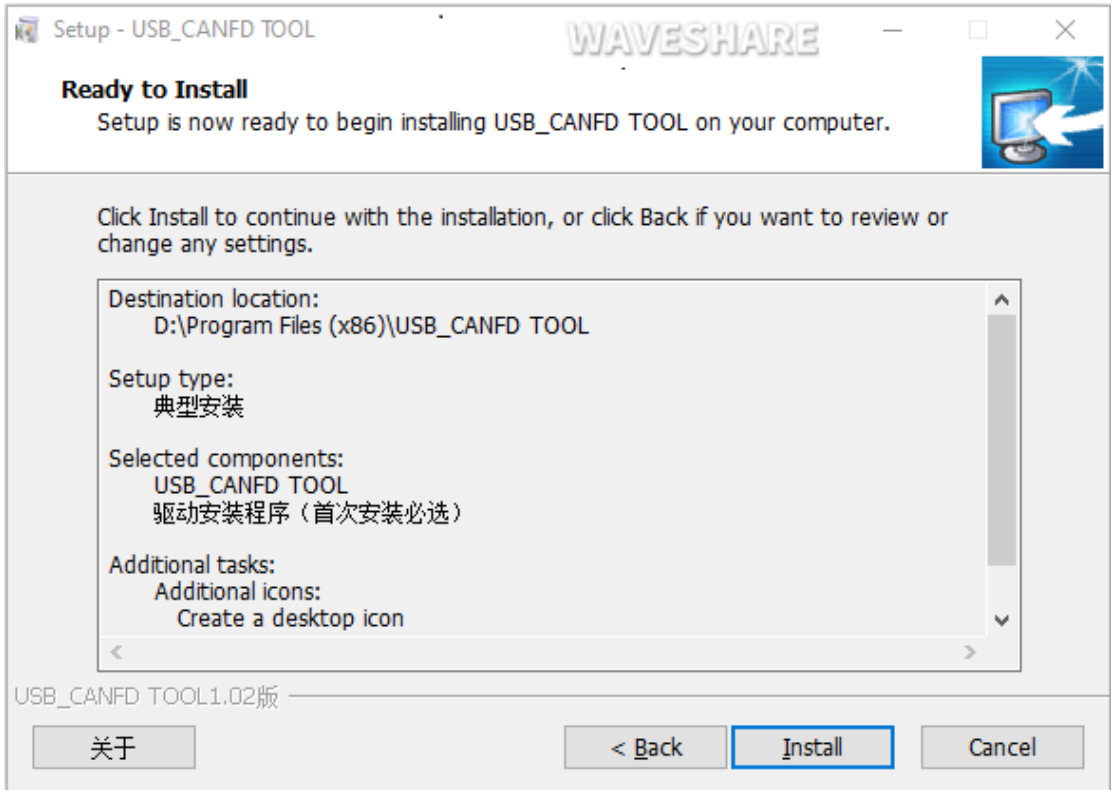
(/wiki/File:USB-CAN-FD_Tool02.png)

- It is recommended to check the box to create a desktop icon and continue to click Next



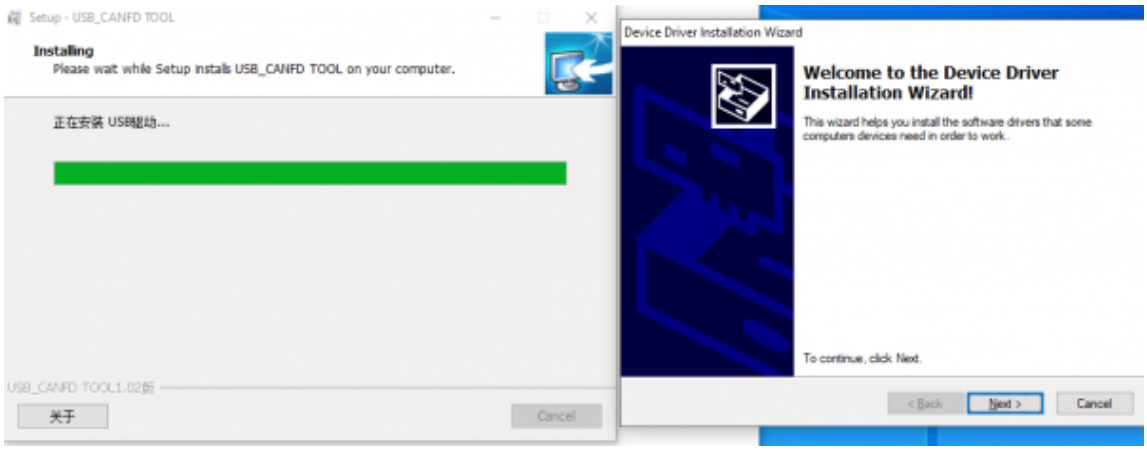
(/wiki/File:USB-CAN-FD_Tool03.png)

- Click Install



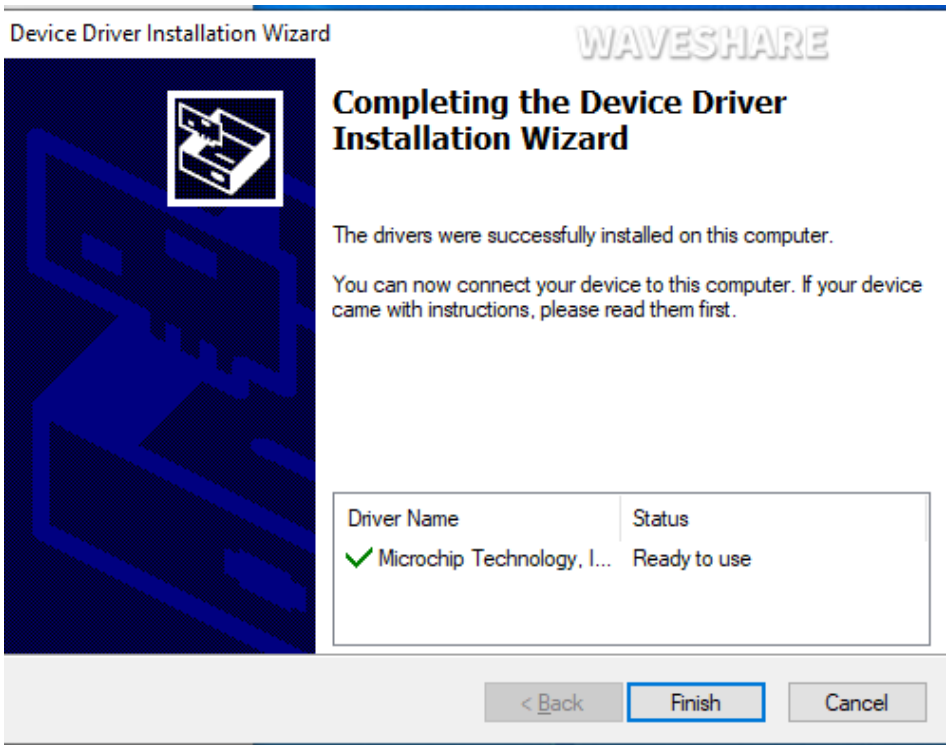
(/wiki/File:USB-CAN-FD_Tool04.png)

- After completion, continue to click Next



(/wiki/File:USB-CAN-FD_Tool05.png)

- After completing the installation, CANFDToolPro will appear on the desktop and in the applications



(/wiki/File:USB-CAN-FD_Tool06.png)

How to Use

> If the communication is abnormal, set the two 120Ω termination resistors of CAN1/CAN2 to ON.

Connect the USB port of the device to the USB port of the PC via the USB cable. Run USB-CAN-FD_TOOL.exe, the following is the description of common functions.

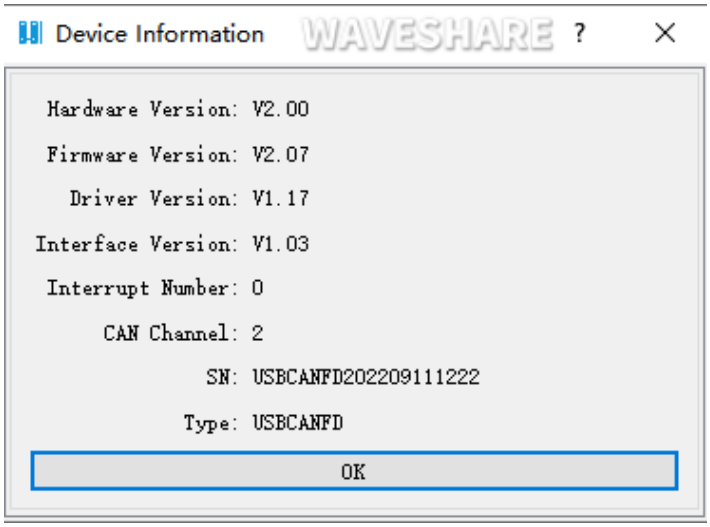
1. Device Management

Select Device Manage(O) -> Open Device to open the device management interface. Here you can open the device, start and stop the channel, set parameters, and other related operations.



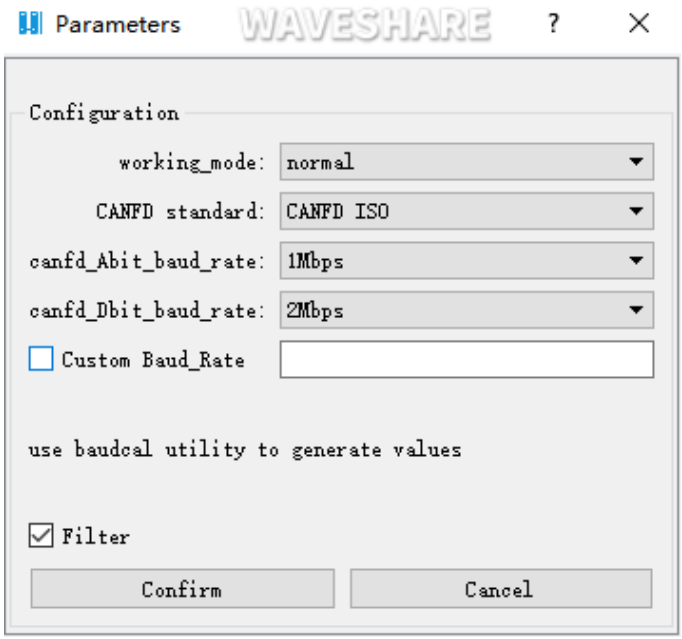
(/wiki/File:USB-CAN-FD_Dev01.png)

- Click Device Information to view software and hardware version information of the current device.



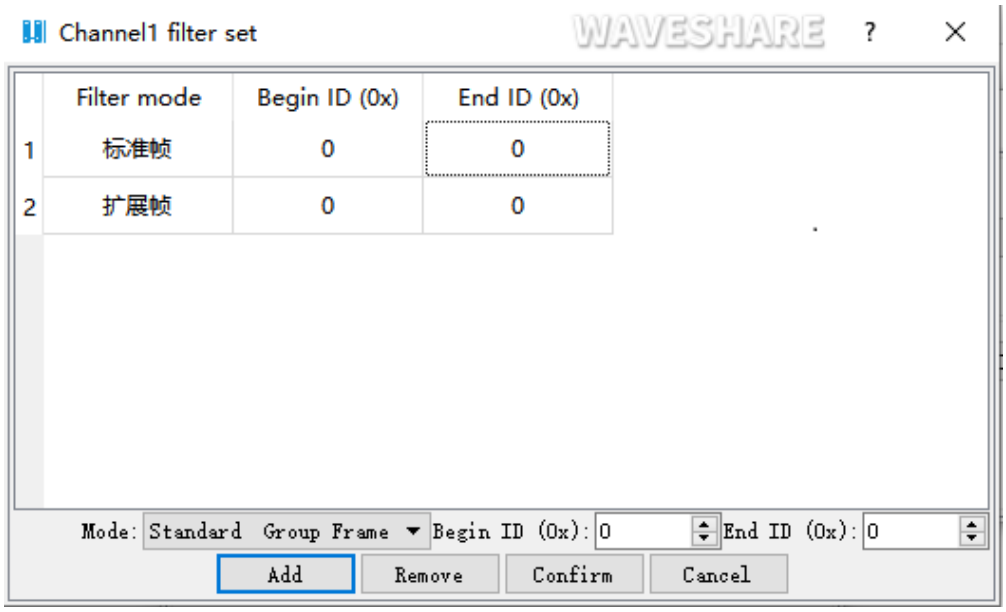
(/wiki/File:USB-CAN-FD_Dev02.png)

- When starting the channel, the parameter setting interface is as follows. You can select the working mode, CANFD standard, arbitration baud rate, data baud rate, and so on.



(/wiki/File:USB-CAN-FD_Dev03.png)

- The filter settings are shown below. Note that up to 64 filter groups can be set for standard frames and up to 32 filter groups can be set for extended frames.

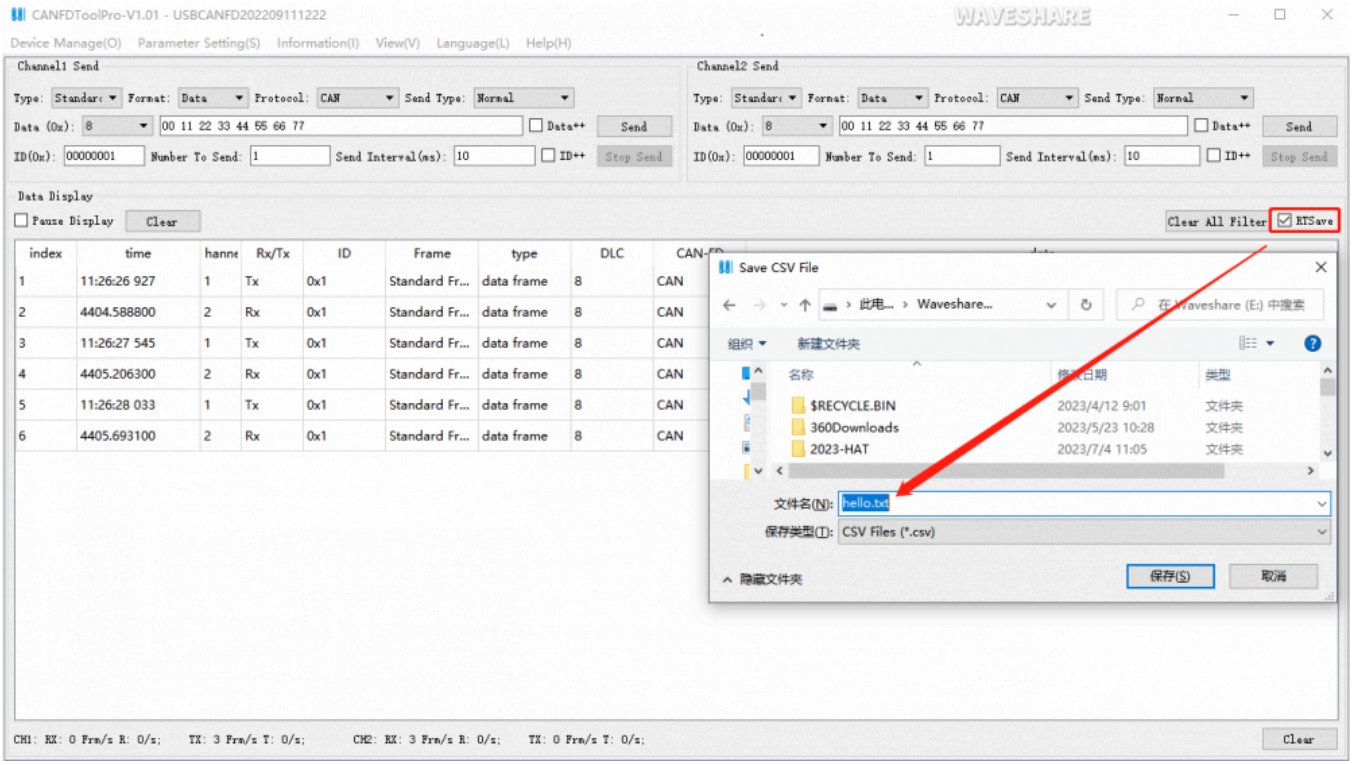


(/wiki/File:USB-CAN-FD_Dev04.png)

2. Display Function

2.1 Save in Real Time

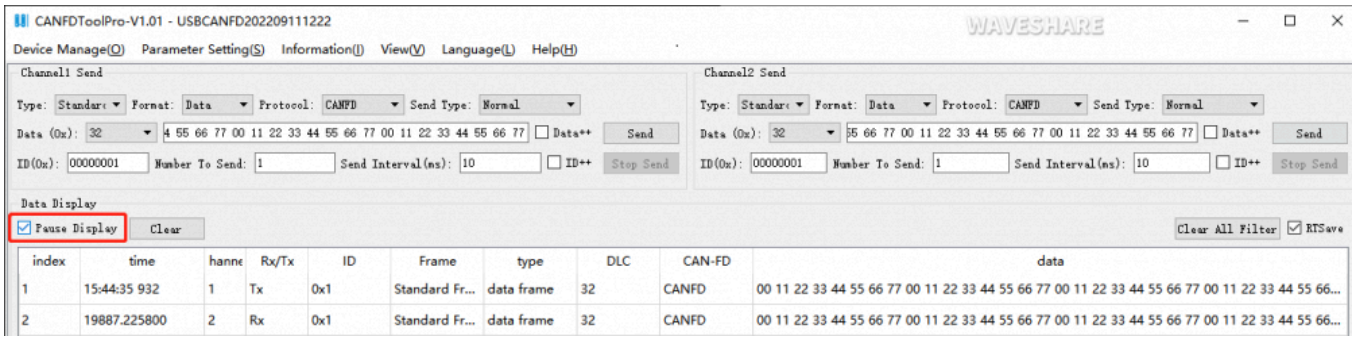
- The default save format is csv, if you want to save as txt format, then directly enter the .txt suffix.



(/wiki/File:USB-CAN-FD_Dev05.png)

2.2 Pause Display

- When receiving data, you can check Pause display for the convenience of viewing the data (at this time, the data is still being received, just not displayed in the current interface). Uncheck the box and the data will continue to be displayed.



(/wiki/File:USB-CAN-FD_Dev06.png)

2.3 Transmission Counts

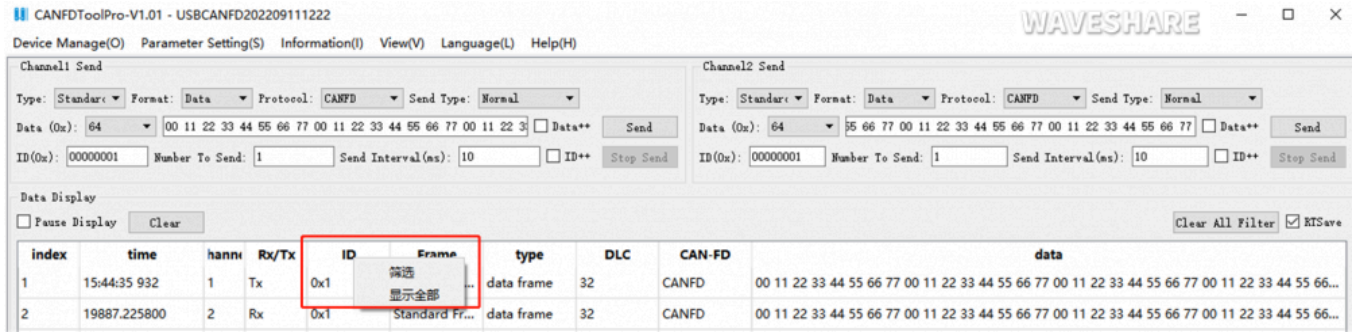
- At the bottom of the software, the total number of receipts and transmissions as well as the real-time frame rate of receipts and transmissions for the 2 channels are counted. Press the Clear button to clear the count.



(/wiki/File:USB-CAN-FD_De07.png)

2.4 Display Buffer

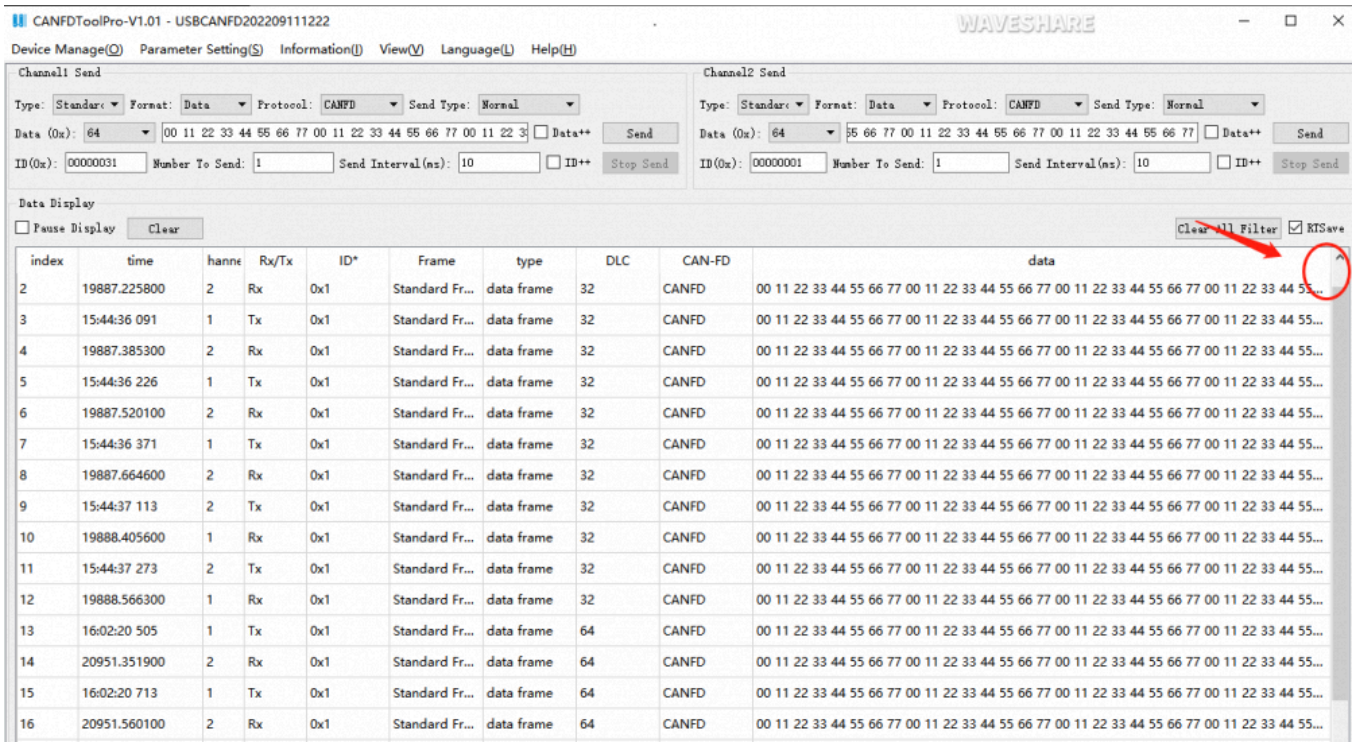
- Display Buffer indicates the maximum number of frames that can be displayed in the current interface, the default value is 1000.



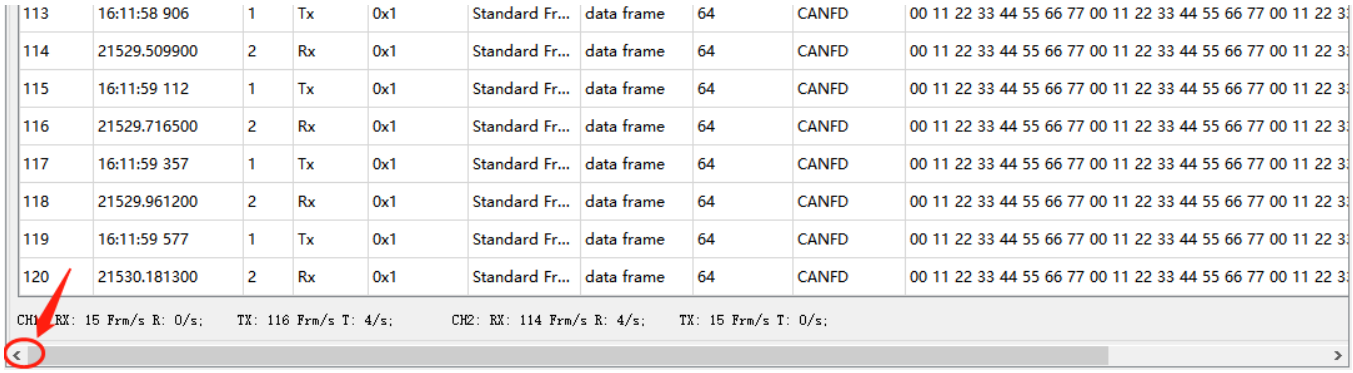
(/wiki/File:USB-CAN-FD_Dev07.png)

2.5 View the Data

- When the data is too long (e.g. 64 bytes), the interface can not display a complete frame of data, put the mouse to the following position, when it becomes a two-way symbol, double-click, and you can expand all the data content, drag the horizontal scroll bar below to view the data.



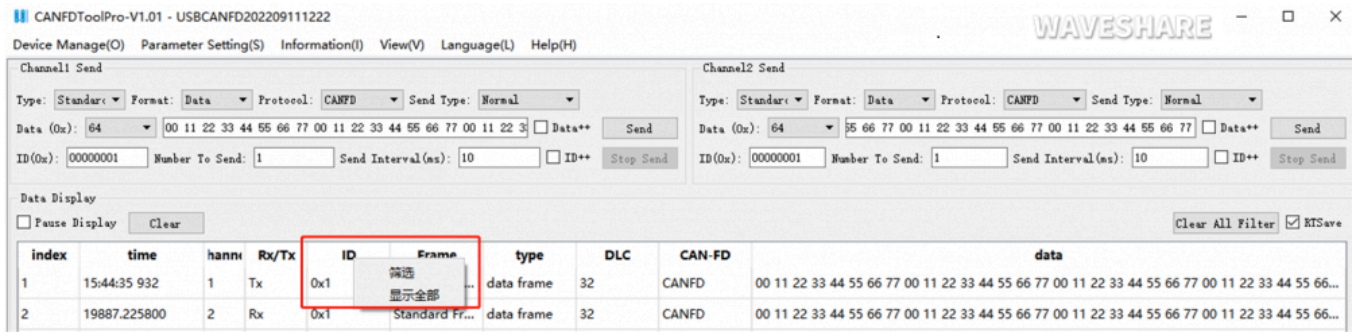
(/wiki/File:USB-CAN-FD_Dev09.png)



(/wiki/File:USB-CAN-FD-bottom.png)

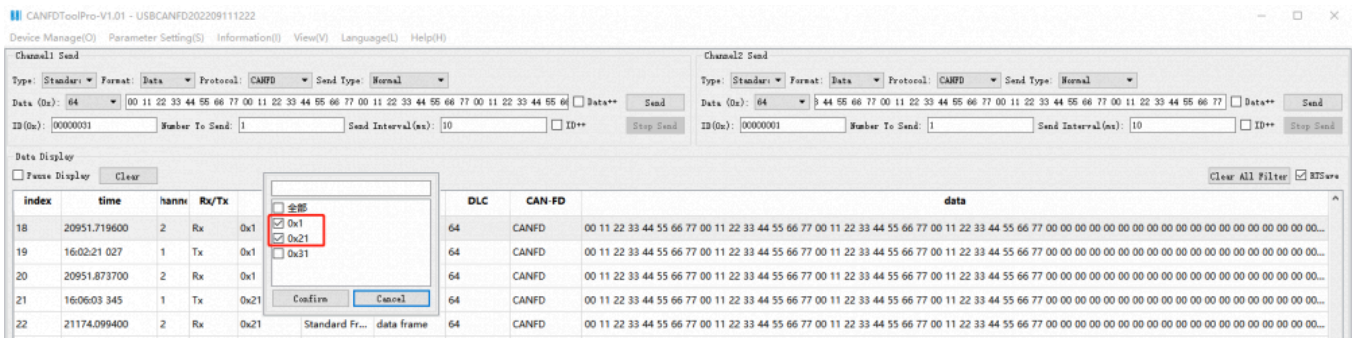
3. Filter Function

- In the data display box, all the displayed items can be filtered conditionally. Right-click to select the item to be filtered, and the filtering menu will pop up, click on Filter to pop up all the contents of the item that can be filtered; click on Show All to cancel the filter conditions.



(/wiki/File:USB-CAN-FD_Dev109.png)

- The filter options will list all different values, and you can check one or more filter conditions.



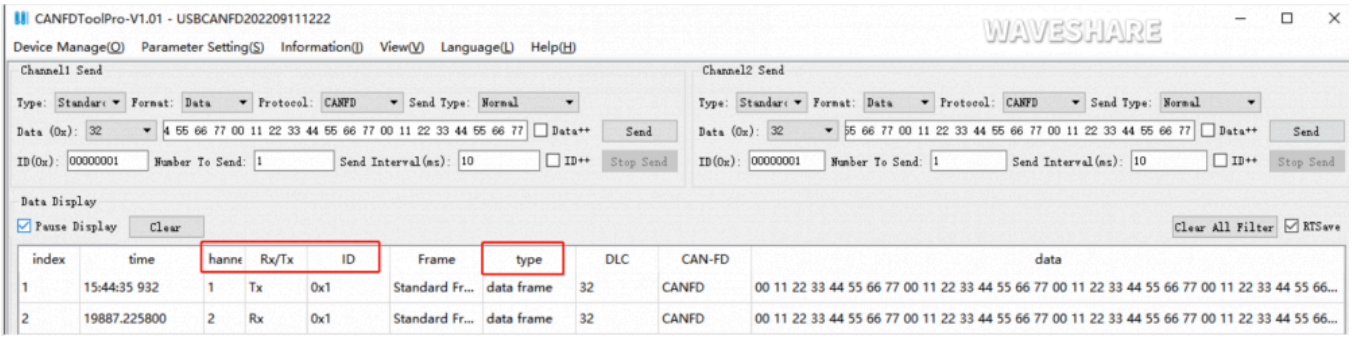
(/wiki/File:USB-CAN-FD_Dev111.png)

- Items that have been set up with filter conditions will be prompted with a *, such as ID set up with filter conditions, it will be displayed as ID*.

index	time	hanne	Rx/Tx	ID*	Frame	type	DLC	CAN-FD	data
6	19887.520100	2	Rx	0x1	Standard Fr...	data frame	32	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
7	15:44:36 371	1	Tx	0x1	Standard Fr...	data frame	32	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
8	19887.664600	2	Rx	0x1	Standard Fr...	data frame	32	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
9	15:44:37 113	2	Tx	0x1	Standard Fr...	data frame	32	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
10	19888.405600	1	Rx	0x1	Standard Fr...	data frame	32	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
11	15:44:37 273	2	Tx	0x1	Standard Fr...	data frame	32	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
12	19888.566300	1	Rx	0x1	Standard Fr...	data frame	32	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
13	16:02:20 505	1	Tx	0x1	Standard Fr...	data frame	64	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
14	20951.351900	2	Rx	0x1	Standard Fr...	data frame	64	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
15	16:02:20 713	1	Tx	0x1	Standard Fr...	data frame	64	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
16	20951.560100	2	Rx	0x1	Standard Fr...	data frame	64	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
17	16:02:20 873	1	Tx	0x1	Standard Fr...	data frame	64	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
18	20951.719600	2	Rx	0x1	Standard Fr...	data frame	64	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
19	16:02:21 027	1	Tx	0x1	Standard Fr...	data frame	64	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
20	20951.873700	2	Rx	0x1	Standard Fr...	data frame	64	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
21	16:06:03 345	1	Tx	0x21	Standard Fr...	data frame	64	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
22	21174.099400	2	Rx	0x21	Standard Fr...	data frame	64	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77
23	16:06:03 560	1	Tx	0x21	Standard Fr...	data frame	64	CANFD	00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77

(/wiki/File:USB-CAN-FD_Dev112.png)

- All the filter conditions are "and" relationship. For example, to view data frames with ID 0x55 for channel 1, you need to set the filter conditions as follows: Channel -> Select 1, Rx/Tx -> Select Rx, ID -> Select 0x55, type -> Select data frame. As shown in the figure below:



(/wiki/File:USB-CAN-FD_Dev113.png)

- If you want to clear all the set filter conditions, click Clear All Filter button.

Loopback Test

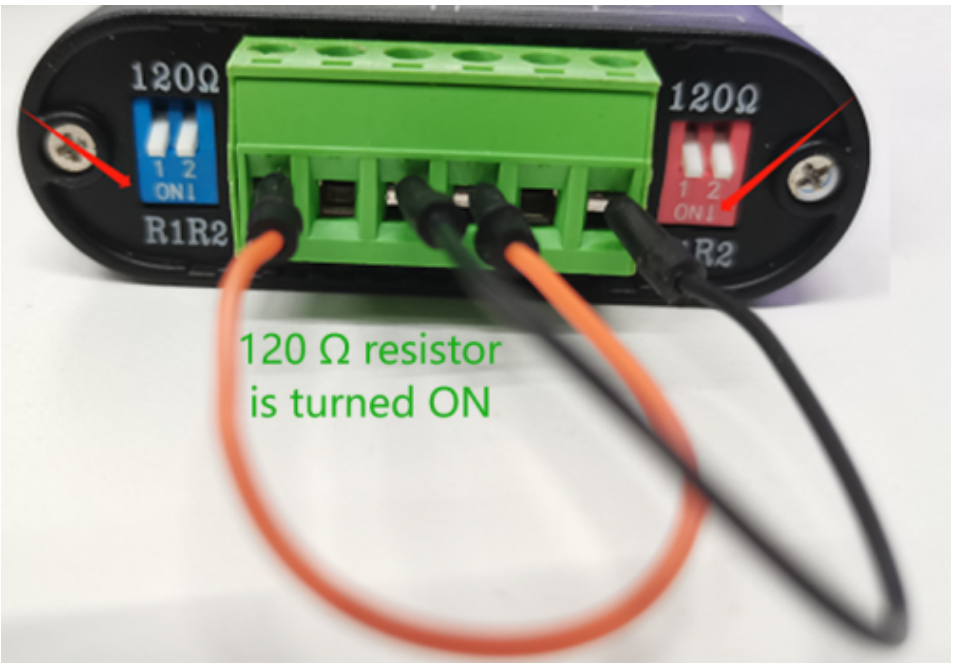
The loopback test is a reliable way to view hardware function, and it's the preferred method for introductory testing. The key steps are as follows:

1. Connect the H and L of channel one to the H and L of channel two correspondingly; keep

the 120-ohm resistors on both channels set to ON (if set to OFF, one channel sends data while the other channel fails to receive it), as shown in the figure below:

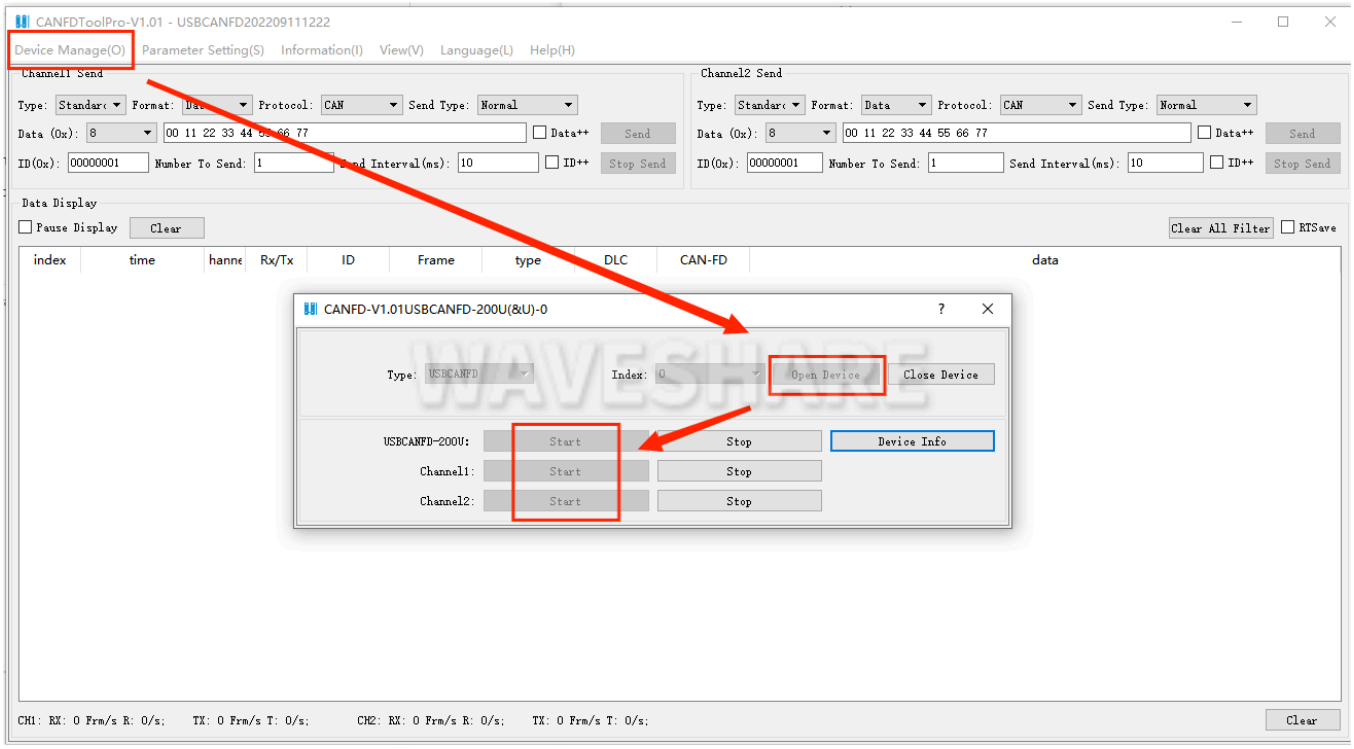


(/wiki/File:USB-CAN-FD_test00.png)



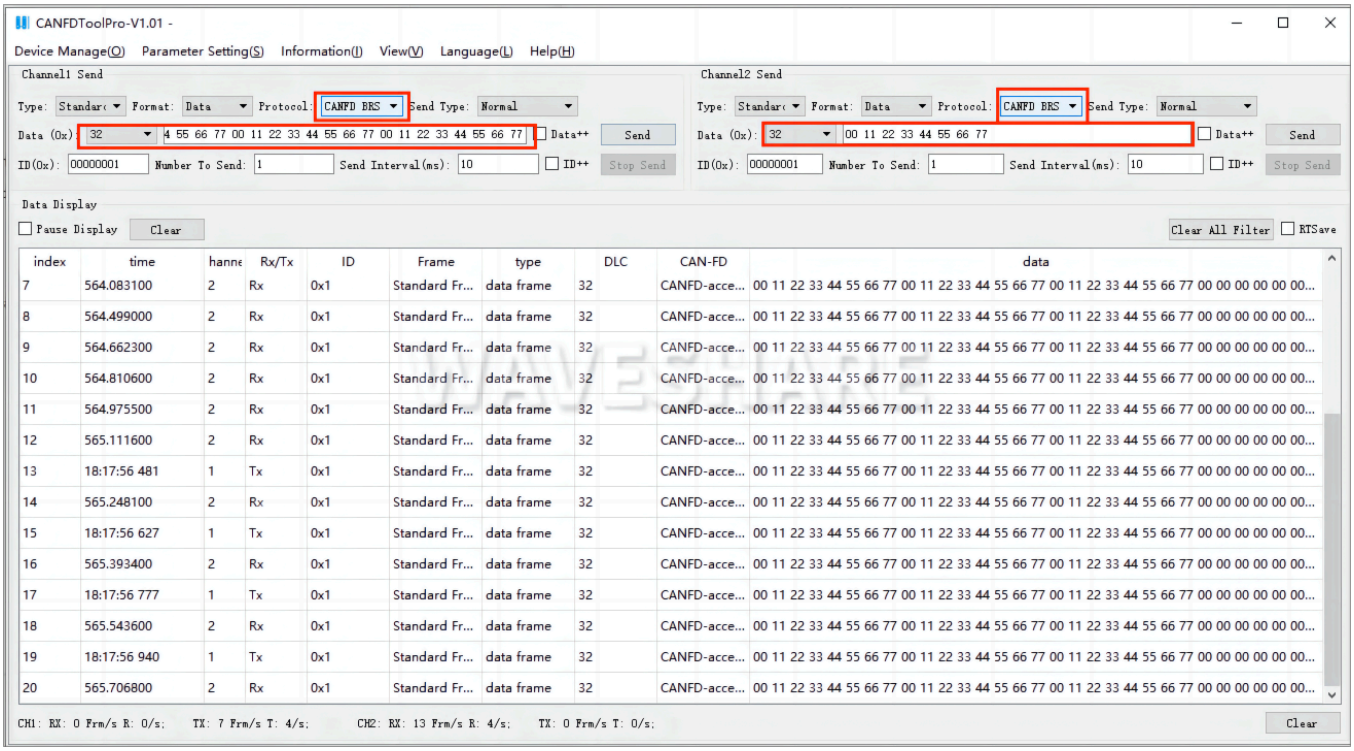
(/wiki/File:USB-CAN-FD_test01.png)

2. Open CANFDTool, click Device Manage(O) -> Open Device -> Start.



(/wiki/File:USB-CAN-FD_test02.png)

3. Set Data (0x) to 32, Protocol to CANFD BRS, send message 00 11 22 33 44 55 66 77 00 11 22 33 44 55 66 77 -> Receive corresponding message.



(/wiki/File:USB-CAN-FD_test03.png)

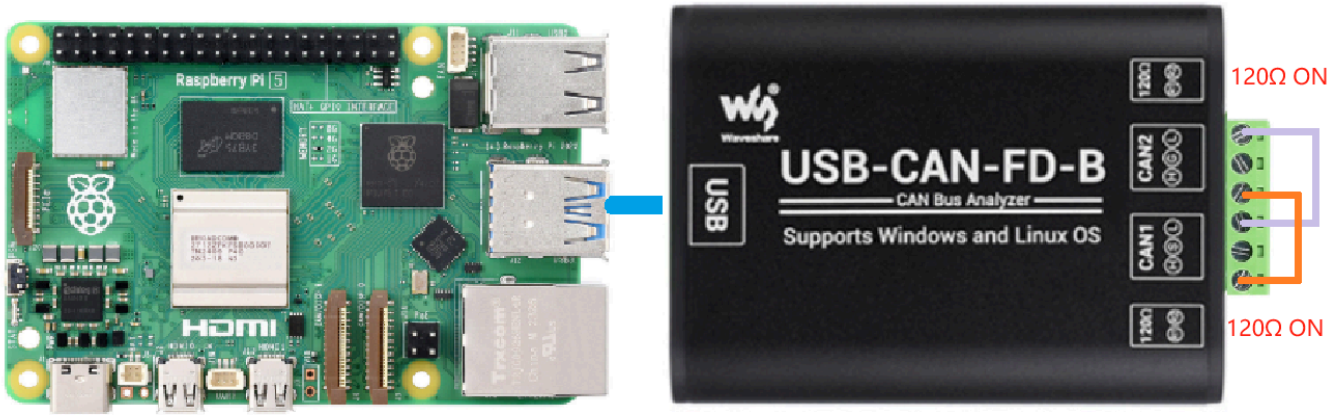
Linux Tutorial

> The Linux tutorial is suitable for USB-CAN-FD-B. USB-CAN-FD does not support Linux system and you can ignore this part.

Raspberry Pi OS

USB-CAN-FD-B can be used on mainstream 32-bit Linux motherboards, taking Raspberry Pi 4B as an example.

- Connect USB-CAN-FD-B and Raspberry Pi via USB, and connect the H and L interfaces of USB-CAB-FD-B channel one to the H and L interfaces of channel two respectively, as shown in the figure below:



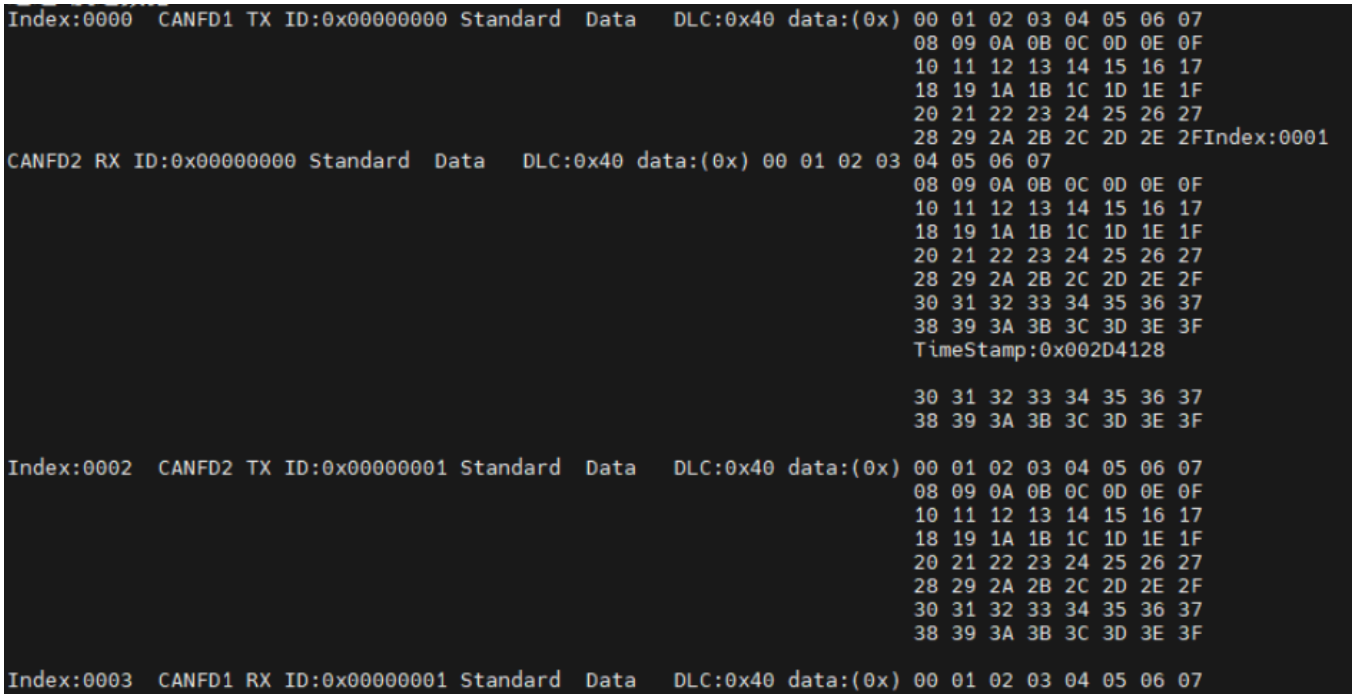
(/wiki/File:USB-CAN-FD-B-rpi.png)

- Copy the terminal command lines of the following c or python demo to Raspberry Pi Terminal to run:

c

```
cd ~
wget https://files.waveshare.com/wiki/USB-CAN-FD/USB-CAN-FD-B-Linux.zip
unzip USB-CAN-FD-B-Linux.zip
sudo chmod 777 -R USB-CAN-FD-B-Linux
cd /USB-CAN-FD-B-Linux/Raspberry/c
sudo make clean
sudo make
sudo ./hello_cpp
```

- After running the C demo, the results are as follows:



(/wiki/File:USB-CAN-FD-B-c.png)

python

```
cd ~
cd /USB-CAN-FD-B-Linux/Raspberry/python3/
sudo python3 python3.8.0.py
```

The results of running the python demo are as follows:

```
jetson@ubuntu:~/Music/python3.8.0$ sudo python3  python3.8.0.py
#####################################################
## Chuang Xin USBCANFD python(x64) test program V2.0 ###
#####################################################
./libcontrolcanfd.so
ZCAN_OpenDevice Enter
open_usb_device ret:0
Open Device OK, device handle:0xffffabdf6d18.
Set CAN0 abit:1M OK!
Set CAN0 dbit:5M OK!
Set CAN1 abit:1M OK!
Set CAN1 dbit:5M OK!
Set CAN0 ISO mode OK!
Set CAN1 ISO mode OK!
Init CAN0 OK!
ZCAN_StartCAN can_ind:0
begin pthread_create pthread_0
Start CAN0 OK!
my_recv_thread dev_idx:0
my_recv_thread can_idx:0
my_recv_thread begin while!
Init CAN1 OK!
ZCAN_StartCAN can_ind:1
begin pthread_create pthread_1
Start CAN1 OK!
my_recv_thread dev_idx:0
my_recv_thread can_idx:1
my_recv_thread begin while!

 CAN0 Tranmit CANFD Num: 10.
CAN1 Received CANFD NUM: 1.
[0]:ts:157016400, id:5, len:16, eff:1, rtr:0, esi:0, brs: 1, data:0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

 CAN0 Tranmit CAN Num: 10.
CAN1 Received CAN NUM: 2.
[0]:ts:157018700, id:5, len:8, eff:1, rtr:0, data:0 1 2 3 4 5 6 7
[1]:ts:157018800, id:6, len:8, eff:1, rtr:0, data:0 1 2 3 4 5 6 7
```

(/wiki/File:Usb-can-fd-b-jetson.png)

Ubuntu (Jetson Nano)

c

```
wget https://files.waveshare.com/wiki/USB-CAN-FD/USB-CAN-FD-B-Linux.zip
unzip USB-CAN-FD-B-Linux.zip
sudo chmod 777 -R USB-CAN-FD-B-Linux
cd /VMware/x86-c
sudo make clean
sudo make
sudo ./hello_cpp
```

python

```
cd ~
cd /USB-CAN-FD-B-Linux/VMware/x86-python3/
sudo python3 python3.8.0.py
```

Linux for VMware virtual machines

c

```
wget https://files.waveshare.com/wiki/USB-CAN-FD/USB-CAN-FD-B-Linux.zip
unzip USB-CAN-FD-B-Linux.zip
sudo chmod 777 -R USB-CAN-FD-B-Linux
cd USB-CAN-FD-B-Linux/VMware/x86-c/
sudo make clean
sudo make
sudo ./hello_cpp
```

```
Index:0000  CANFD1 TX ID:0x00000000 Standard  Data   DLC:0x40 data:(0x) 00 01 02 03 04 05 06 07
                                                                        08 09 0A 0B 0C 0D 0E 0F
                                                                        10 11 12 13 14 15 16 17
                                                                        18 19 1A 1B 1C 1D 1E 1F
                                                                        20 21 22 23 24 25 26 27
                                                                        28 29 2A 2B 2C 2D 2E 2F
                                                                        30 31 32 33 34 35 36 37
                                                                        38 39 3A 3B 3C 3D 3E 3F

Index:0001  CANFD2 TX ID:0x00000001 Standard  Data   DLC:0x40 data:(0x) 00 01 02 03 04 05 06 07
                                                                        08 09 0A 0B 0C 0D 0E 0F
                                                                        10 11 12 13 14 15 16 17
                                                                        18 19 1A 1B 1C 1D 1E 1F
```

(/wiki/File:USB-CAN-FD-B-Linux-vmware.png)

python

```
cd ~
cd USB-CAN-FD-B-Linux/VMware/x86-python3/
sudo python3 python3.8.0.py
```

```
h@ubuntu:~/USB-CAN-FD-B-Linux/VMware/x86-python3$ sudo python3 python3.8.0.py
#######################################################
## Chuang Xin USBCANFD python(x64) test program V2.0 ###
#######################################################
./libcontrolcanfd.so
ZCAN_OpenDevice Enter
open_usb_device ret:0
Open Device OK, device handle:0x7fc99a423d18.
Set CAN0 abit:1M OK!
Set CAN0 dbit:5M OK!
Set CAN1 abit:1M OK!
Set CAN1 dbit:5M OK!
Set CAN0 ISO mode OK!
Set CAN1 ISO mode OK!
Init CAN0 OK!
ZCAN_StartCAN can_ind:0
begin pthread_create pthread_0
Start CAN channel OK!
my_recv_thread dev_idx:0
my_recv_thread can_idx:0
my_recv_thread begin while!
Init CAN1 OK!
ZCAN_StartCAN can_ind:1
begin pthread_create pthread_1
my_recv_thread dev_idx:0
my_recv_thread can_idx:1
Start CAN channel OK!
my_recv_thread begin while!

CAN0 Transmit CANFD Num: 10.
CAN1 Received CANFD NUM: 2.
[0]:ts:22328719000, id:5, len:16, eff:1, rtr:0, esi:0, brs:1, data:0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
[1]:ts:22328722200, id:6, len:16, eff:1, rtr:0, esi:0, brs:1, data:0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

CAN0 Transmit CAN Num: 10.
CAN1 Received CAN NUM: 2.
[0]:ts:22328740000, id:5, len:8, eff:1, rtr:0, data:0 1 2 3 4 5 6 7
[1]:ts:22328744000, id:6, len:8, eff:1, rtr:0, data:0 1 2 3 4 5 6 7
my_recv_thread end return!
Close CAN0 OK!
my_recv_thread end return!
Close CAN1 OK!
close_usb_device index:0
libusb_close return
Close Device OK!
```

(/wiki/File:USB-CAN-FD-B-Vmware-python.png)

Resources

Windows Driver and Tool

- USB-CAN-FD/USB-CAN-FD-B Windows Driver (https://files.waveshare.com/wiki/USB-CAN-FD/Driver/driver_en.zip)
- USB-CAN-FD/USB-CAN-FD-B Windows Debugging Tool (https://files.waveshare.com/wiki/USB-CAN-FD/tool/USB_CANFD%20TOOLSetup(V1.16).exe.zip)

- Secondary Development Library Files (https://files.waveshare.com/wiki/USB-CAN-FD/Demo/USB-CAN-FD_%20Library.zip)

Linux Driver Demo (Only applicable to USB-CAN-FD-B)

- USB-CAN-FD-B Linux demo (https://files.waveshare.com/wiki/USB-CAN-FD/Demo/USB-CAN-FD-B-Linux.zip)

Document

- Secondary Development API Library Guide (https://files.waveshare.com/wiki/USB-CAN-FD/Manual/USB-CAN-FD-B-API-Manual.pdf)

FAQ

Question: Why do the 120-ohm resistors for each channel have two switches, and what is the difference between turning on one and two switches?

Answer:
Only turn on a 120Ω switch for one channel, so a 120Ω resistor is connected in series between H and L, and then turn on another switch, which means another 120Ω resistor is connected in parallel, so the resistance between H and L is 60Ω.

Question: Why is the communication with the external CAN device abnormal, while the two-channel loopback test is okay?

Answer:

- Look at the arbitration baud rate and data baud rate configuration of CAN. If the external device is an ordinary CAN, 500K, it is set to 500K. If it is CAN FD, you may need to configure the arbitration baud rate and data baud rate separately. The difference in baud rate between ordinary CAN and CAN FD is as follows:

Normal CAN:

- Arbitration baud rate: When sending data, nodes decide who has priority through an arbitration process, usually set to a unified baud rate (such as 1 Mbps).
- Data baud rate: In normal CAN, the arbitration baud rate and data baud rate are usually the same.

CAN FD:

- Arbitration baud rate: Similar to normal CAN, the arbitration baud rate is used to determine priority, usually a lower rate (such as 1 Mbps).
- Data baud rate: It can be higher than the arbitration baud rate and supports higher data transfer rates (up to 8 Mbps) to improve data transmission efficiency.

Support

Technical Support

If you need technical support or have any feedback/review, please click the **Submit Now** button to submit a ticket, Our support team will check and reply to you within 1 to 2 working days. Please be patient as we make every effort to help you to resolve the issue.
Working Time: 9 AM - 6 PM GMT+8 (Monday to Friday)

Submit Now (https://service.waveshare.com/)

Retrieved from "https://www.waveshare.com/w/index.php?title=USB-CAN-FD&oldid=107972 (https://www.waveshare.com/w/index.php?title=USB-CAN-FD&oldid=107972)"