USB-CAN-A

From Waveshare Wiki

Jump to: navigation, search

USB-CAN-A is a USB-serial-CAN solution. In almost all application scenarios, the serial port baud rate defaults to 2M and does not need to be changed!!!

USB-CAN-A

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

USB To CAN, CAN

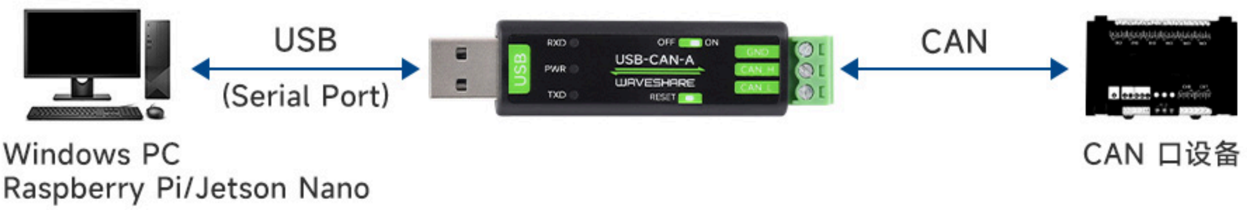
Overview

Features

- Supports CAN2.0A (standard frame) and CAN2.0B (extended frame)
- Supports CAN communication baud rate range: 5kbps~1Mbps, customizable CAN baud rate
- Supports four working modes: normal mode, loopback mode, silent mode, and silent loopback mode
- Supports multiple CAN data transmission methods: single-frame transmission, multi-frame transmission, manual transmission, timed transmission, and cyclic transmission
- Supports multiple CAN data reception methods: can receive only data with specified IDs, or configure IDs to automatically respond with predefined data
- Supports transmitting and receiving CAN data and save it as TXT or EXCEL text
- Supports CAN bus detection, making it easy to check the CAN bus status
- CAN transmitting and receiving data display is marked with time stamps, which can be displayed sequentially or refreshed
- Supports setting the communication baud rate of the USB virtual COM port (9600 ~ 2000000bps), which is 2000000bps by default
- Supports setting operation parameters via configuration software or serial port commands, capable of saving settings after power off
- Adopts imported STM32 chip solution, stable and reliable communication
- Onboard TVS (Transient Voltage Suppressor) effectively suppresses surge voltage and transient spike voltage in the circuit
- Provides both Chinese and English versions of the upper computer software for Windows, easy to use

- Supports secondary development, only need to process transmitting and receiving commands

Primary Functions

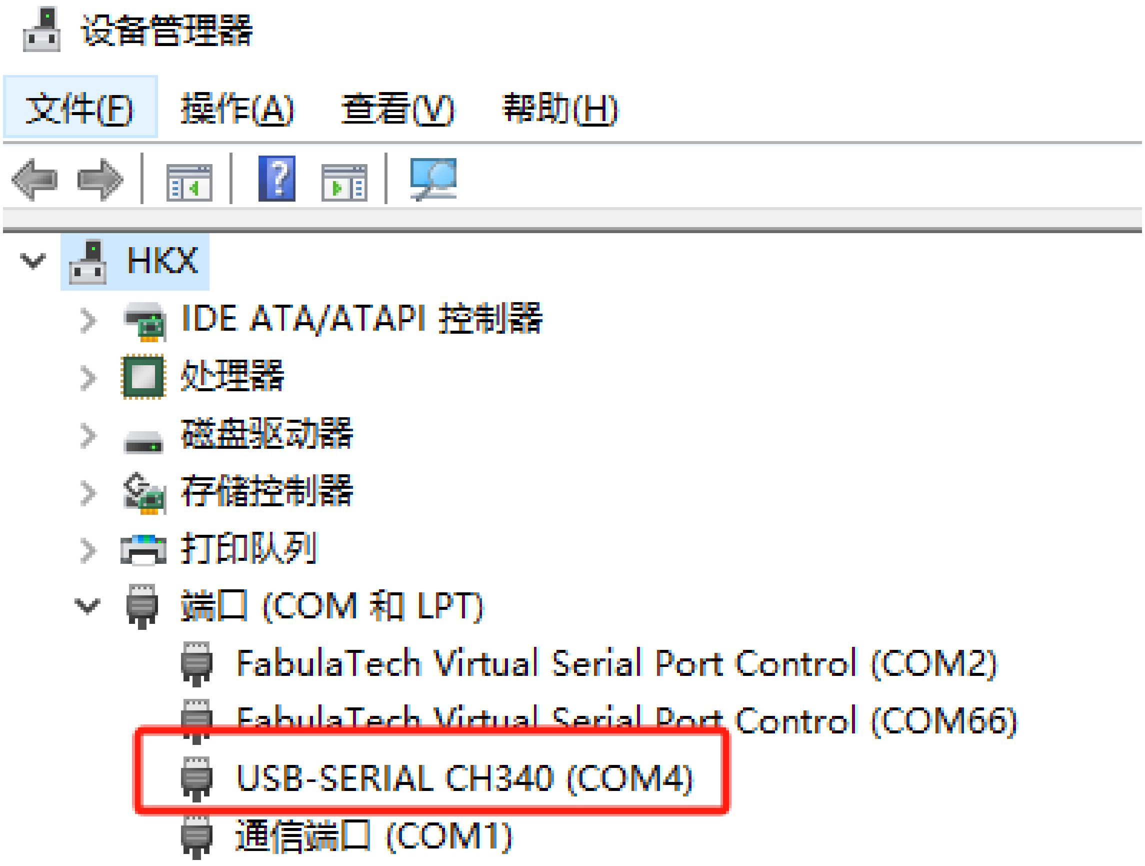


(/wiki/File:USB-CAN-A-details-5.jpg)

Working with Windows

Install USB Driver

- Download CH341SER driver (https://files.waveshare.com/upload/c/c4/CH341SER.zip), double click "CH341SER.EXE" file, and click Install. After the "Driver Installation Successful" message appears, click OK to complete the driver installation:

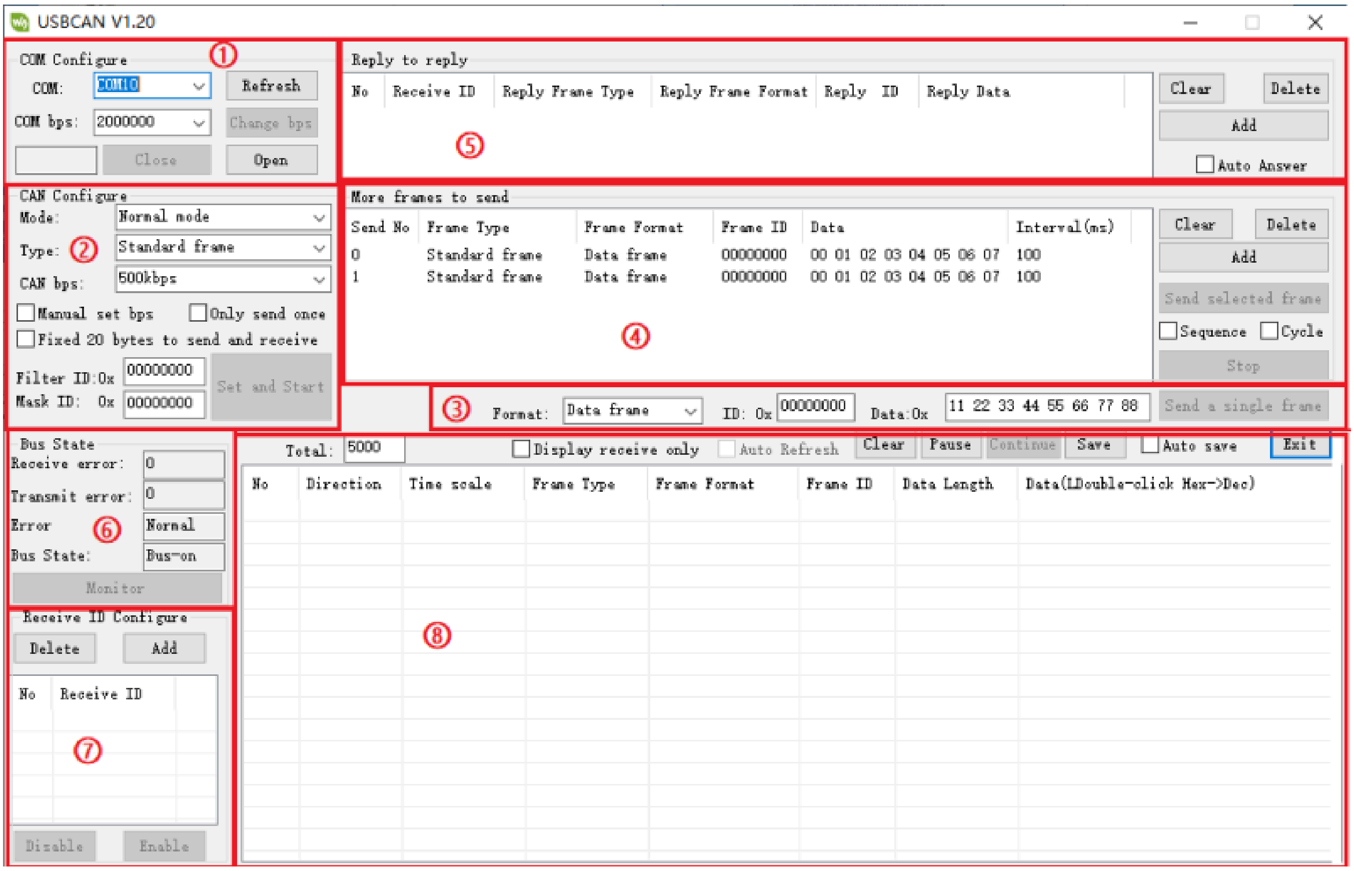

设备管理器
文件(F) 操作(A) 查看(V) 帮助(H)
HKX
IDE ATA/ATAPI 控制器
处理器
磁盘驱动器
存储控制器
打印队列
端口 (COM 和 LPT)
FabulaTech Virtual Serial Port Control (COM2)
FabulaTech Virtual Serial Port Control (COM66)
USB-SERIAL CH340 (COM4)
通信端口 (COM1)

(/wiki/File:USB-CAN-A-serial.png)

CAN Configuration Tool User Guide

How to Use CAN Configuration Tool

- This tool does not require installation. Download USB-CAN-A_TOOL_2.12 (https://files.waveshare.com/wiki/USB-CAN-A/Tool/USBCANV2.12_Chinese.zip) and unzip it, then open USB-CAN.exe file:



COM Configure
COM: COM10
COM bps: 2000000

Reply to reply

No	Receive ID	Reply Frame Type	Reply Frame Format	Reply ID	Reply Data

□ Auto Answer

CAN Configure
Mode: Normal mode
Type: Standard frame
CAN bps: 500kbps
□ Manual set bps □ Only send once
□ Fixed 20 bytes to send and receive
Filter ID:0x 00000000
Mask ID: 0x 00000000

More frames to send

Send No	Frame Type	Frame Format	Frame ID	Data	Interval(ms)
0	Standard frame	Data frame	00000000	00 01 02 03 04 05 06 07	100
1	Standard frame	Data frame	00000000	00 01 02 03 04 05 06 07	100

□ Sequence □ Cycle

Format: Data frame ID: 0x 00000000 Data:0x 11 22 33 44 55 66 77 88

Bus State
Receive error: 0
Transmit error: 0
Error: Normal
Bus State: Bus-on

Receive ID Configure

No	Receive ID

Total: 5000 □ Display receive only □ Auto Refresh □ Auto save

No	Direction	Time scale	Frame Type	Frame Format	Frame ID	Data Length	Data(LDouble-click Hex->Dec)

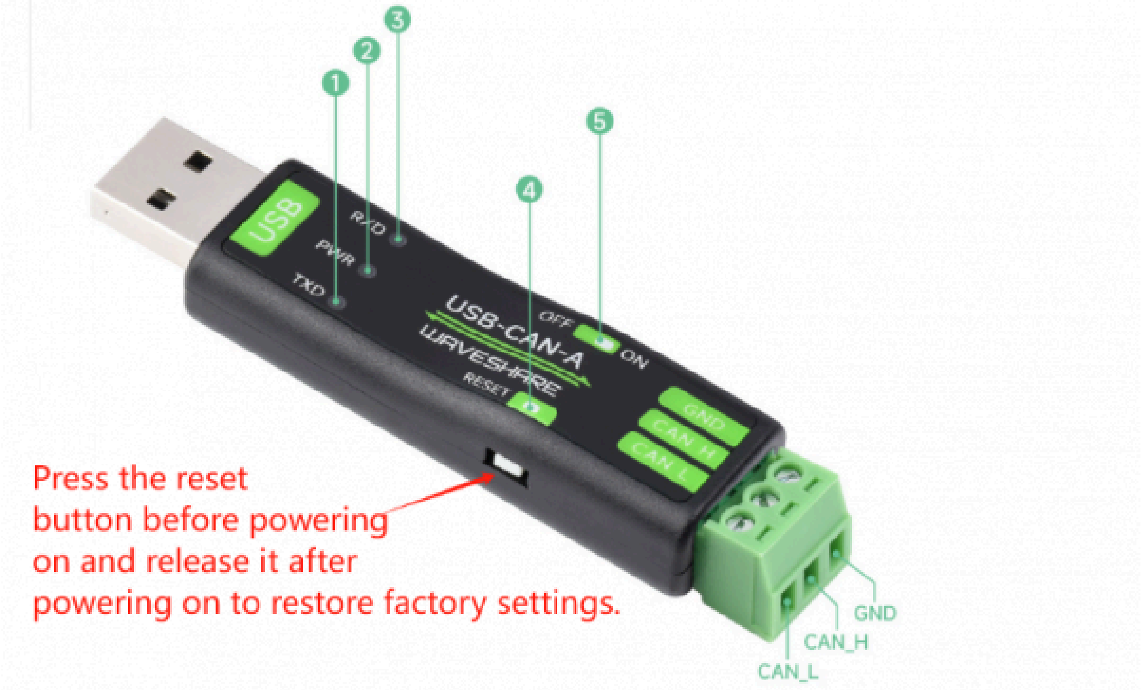
(/wiki/File:800px-USB-CAN-A_WIN01.png)

Function Description of Configuration Page

① COM Configure:

After inserting USB to CAN into the computer and installing the driver, the device manager will display a COM port. The upper computer configuration tool can select the corresponding port of the device manager to open or close the port, similar to the serial port debugging assistant. The default communication baud rate is 2000000bps.

The 2M baud rate is compatible with most CAN rates (the maximum serial port baud rate is set to 2M, and the CAN rate can reach up to 1M). Generally, the default 2M baud rate is sufficient, and the CAN rate can be changed to the rate corresponding to the external CAN device (supporting up to 1M). If another baud rate was previously configured, press the reset button before powering on, release after powering on to restore factory settings and reset the USB-CAN-A to 2M serial port baud rate:


1
2
3
4
5
USB
RXD
PWR
TXD
USB-CAN-A
WAVESHARE
OFF
ON
RESET
GND
CAN_H
CAN_L
Press the reset button before powering on and release it after powering on to restore factory settings.
GND
CAN_H
CAN_L

(/wiki/File:USB-CAN-A-faq.png)

Some serial assistants (using low-level hex data) may have a baud rate lower than 2M. If you need to reduce the baud rate, please refer to the following text:

Instructions for changing the COM(232) baud rate:

1. Connect the converter with a USB data cable, and pay attention to how many times the TX and RX indicators flash:

```
Flash 1 time, the baud rate of the corresponding serial port is 2000000bps
Flash 2 time, the baud rate of the corresponding serial port is 1228800bps
Flash 3 time, the baud rate of the corresponding serial port is 115200bps
Flash 4 time, the baud rate of the corresponding serial port is 38400bps
Flash 5 time, the baud rate of the corresponding serial port is 19200bps
Flash 6 time, the baud rate of the corresponding serial port is 9600bps
```

2. Open the software and choose the corresponding COM port and baud rate, then click the Open button. (If the TX and RX indicators flash 1 time when the converter is connected to the USB cable and powered on, we should choose the baud rate of 2,000,000bps)

3. Click the Open button

4. Click the Change Baud Rate button, and a dialog box will pop up. After we select the baud rate we are going to set in the baud rate combo box, click Change; if you do not want to change, click the Cancel button

5. For example, if we want to set the baud rate as 9600bps:

- Set 9600 in the baud rate combo box
- Click Change button, and observe the RX and TX indicators of the converter
- The RX and TX indicators of the converter corresponding to the 9600 baud rate flashes 6 times
- As the baud rate is changed, you need to close the port and open it again after selecting the changed baud rate

6. Click the Close button

7. Set the baud rate as the changed one

8. Click the Open button again

② CAN Configure:

The operating modes include: normal, loopback, silent, and silent loopback modes

- Normal mode: The normal operating mode of CAN can send and receive the data to and from the bus normally
- Loopback mode: The data can be sent to the CAN bus, and give feedback to the internal reception area, and ignoring the actual state of the receiving pin. This mode is for self-testing.
- Silent mode: It can receive data normally, but can only send implicit bits and cannot actually send messages. It is commonly used to analyze CAN bus activity
- Silent loopback mode: This mode is for "hot testing", which is self-testing. It can be tested like the loopback mode but does not affect the CAN bus system
- Frame type: standard frame (CAN2.0A 11-bit ID), extended frame (CAN2.0B 29-bit ID)
- CAN baud rate: You can directly choose the common ones for CAN communication: 1M, 800K, 500K, 400K, 250K, 200K, 125K, 100K, 50K, 20K, 10K, 5K. You can choose one if the above baud rate that can be directly set is different from your CAN device
- Filter ID and Mask ID: Both are hexadecimal data, the lower 11 bits of filter ID and mask ID are valid in a standard frame (range: 0x00000000~0x000007ff), and the lower 29 bits of filter ID and mask ID in the extended frame are valid (range 0x00000000~0x1fffffff)
- Fixed 20-byte transmission and reception: There are 2 conversion protocols inside the CAN converter, one is a communication protocol with adjustable length, and the other is a fixed 20-byte communication protocol. If selected, it will be a fixed 20-byte communication protocol. If not selected, it will be a variable communication protocol. After selection, a dialog box will pop up to customize the baud rate at the top of which is the CAN baud rate calculation formula. Additionally, you can set phase buffer 1, phase buffer 2, and prescaler
- Disable automatic message retransmission: CAN communication is generally automatically retransmitted if the transmission is unsuccessful. If the data is sent cyclically, you can set it to disable the automatic retransmission of the message. Click Set and start CAN communication

③ Send single-frame data field:

The frame format includes data frame and remote frame, the frame ID is hexadecimal data, the frame ID range for the standard frame is 0x00000000~0x000007fff, and the frame ID range for the extended frame is 0x00000000~0x1fffffff. The data sent is also hexadecimal data, the data in the figure are 0x11 0x22 0x33 0x44 0x55 0x66 0x77 0x88. Click the Send a single frame button to send the frame data to the CAN bus

④ Send multi-frame data field:

- Add: A send frame data will be added below the selected line in the multi-frame sending area
- Delete: delete the chosen line
- Clear: clear all the data in the multi-frame sending data area
- Interval time: the interval sending time of two consecutive frames in the multi-frame sending area (milliseconds)
- Send selected frame: Click this button to send a selected line in the multi-frame area for transmission
- Sequence: It will automatically turn to the next frame after clicking Send selected frame
- Cycle: It will send the selected frame in the cycle after clicking Send selected data

- Stop: cancel the command of sending the multiple frames

⑤ Auto Answer field:

- Add: An auto answer data will be added below the selected line in the auto answer area
- Delete: delete the chosen line
- Clear: clear all the data in the auto answer area
- Auto answer: It will automatically reply to the corresponding frame ID and data after receiving the ID

⑥ CAN Bus Status:

Mainly used to check the CAN communication status of the USB to CAN device

⑦ Receive ID Configure:

- Add: An auto answer will be added under the selected line in the Receive ID Configure area
- Delete: the selected line will be deleted
- Enable: After clicking this button, the converter will only upload the ID set in the Receive ID Configure area, and other IDs will not be uploaded to the computer
- Disable: After clicking this button, the converter will upload all ID data to the computer.

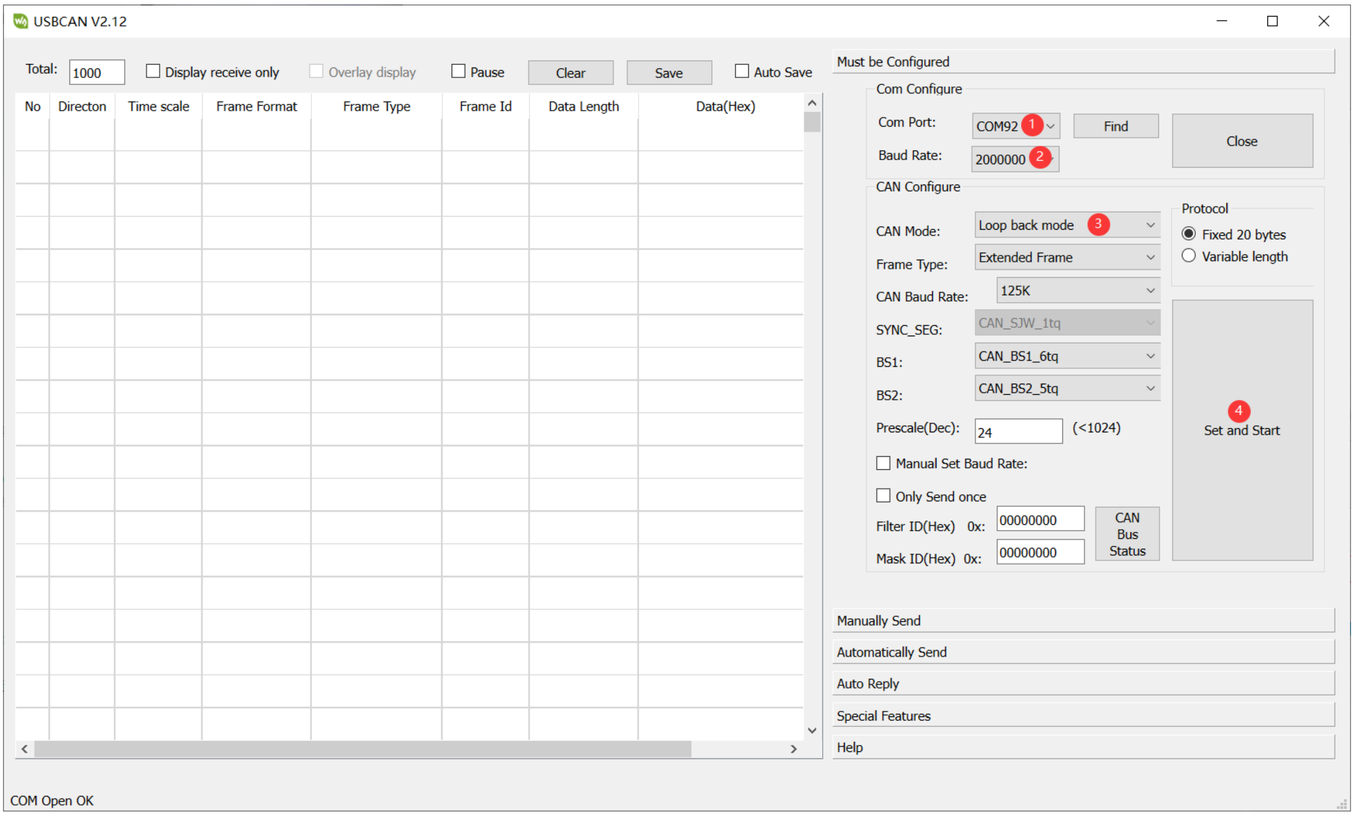
⑧ Send and receive data display field:

- Clear: clear all data on the display for sending and receiving data
- Pause: pause to display other data in the send and receive display area
- Continue: continue to display sending and receiving data
- Save: You can save the data in the sending and receiving data buffer in two formats, excel or text
- Display receive only: only the received data can be displayed in the sending and receiving data area
- Auto Refresh: when only the accepted data is displayed, you can choose to automatically refresh. At this time, the data is refreshed in real-time instead of continuously increasing the column display. This function can summarize the ID data summarized by CAN and select any line

Hardware Test

USB-CAN-A Loopback Mode Test

- First, confirm the baud rate. If the TX and RX lights flash momentarily when powered on, that's correct. Please set the module to loopback mode with a default 2M serial port baud rate (the serial port baud rate is fixed on the computer USB side). The CAN rate and function code should be customized (set to the same rate as the external device, for example, if the motor CAN rate is 1Mbps with IDx123, then set the CAN rate to 1Mbps and ID 0x123). Set and start:



Total: 1000 ☐ Display receive only ☐ Overlay display ☐ Pause ☐ Auto Save

No	Directon	Time scale	Frame Format	Frame Type	Frame Id	Data Length	Data(Hex)

Com Port: COM92

Baud Rate: 2000000

CAN Mode: Loop back mode

Frame Type: Extended Frame

CAN Baud Rate: 125K

SYNC_SEG: CAN_SJW_1tq

BS1: CAN_BS1_6tq

BS2: CAN_BS2_5tq

Prescale(Dec): 24 (<1024)

☐ Manual Set Baud Rate:

☐ Only Send once

Filter ID(Hex) 0x: 00000000

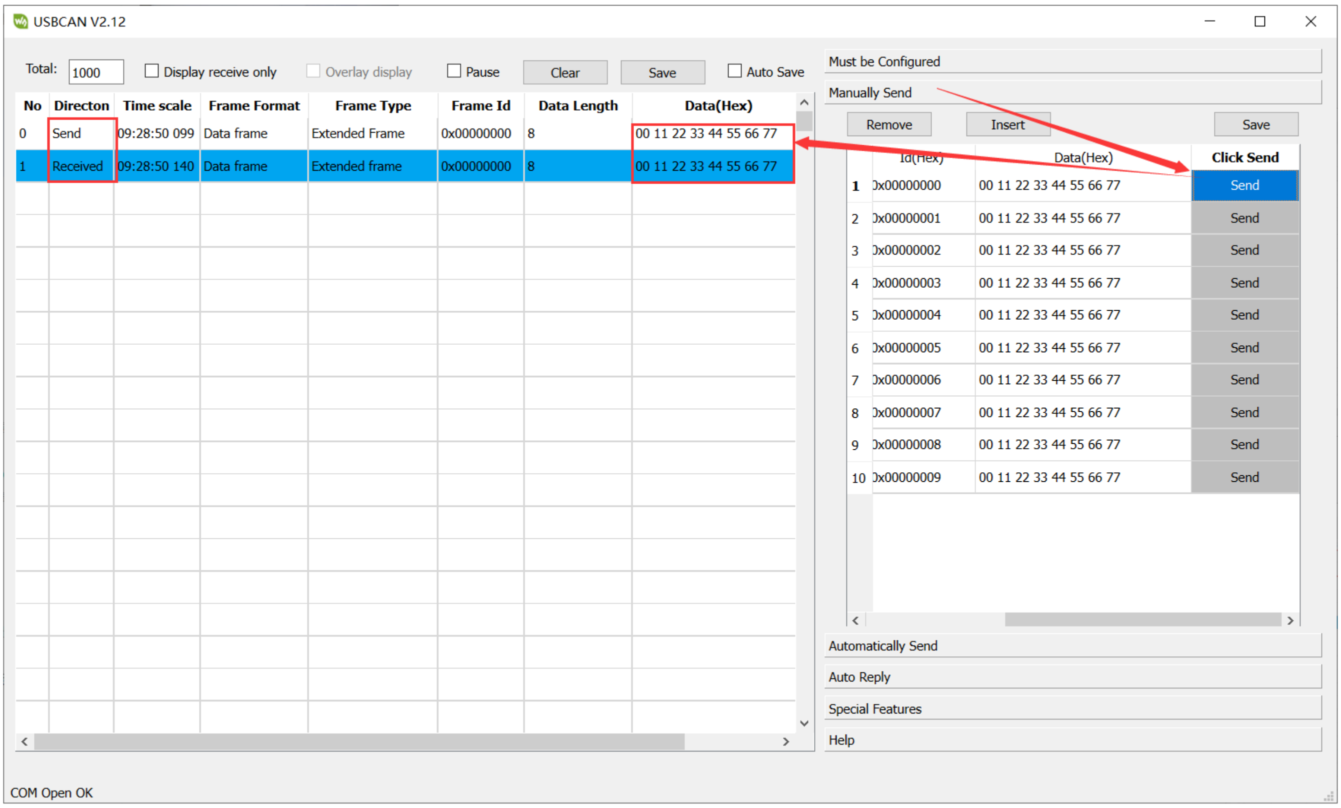
Mask ID(Hex) 0x: 00000000

Protocol
● Fixed 20 bytes
○ Variable length

COM Open OK

(/wiki/File:USB-CAN-A-2.12.png)

- Then click on the Manually Send interface, send the data and accept the same data, the function is OK:



Total: 1000 ☐ Display receive only ☐ Overlay display ☐ Pause ☐ Auto Save

No	Directon	Time scale	Frame Format	Frame Type	Frame Id	Data Length	Data(Hex)
0	Send	09:28:50 099	Data frame	Extended Frame	0x00000000	8	00 11 22 33 44 55 66 77
1	Received	09:28:50 140	Data frame	Extended frame	0x00000000	8	00 11 22 33 44 55 66 77

	Id(Hex)	Data(Hex)	Click Send
1	0x00000000	00 11 22 33 44 55 66 77	Send
2	0x00000001	00 11 22 33 44 55 66 77	Send
3	0x00000002	00 11 22 33 44 55 66 77	Send
4	0x00000003	00 11 22 33 44 55 66 77	Send
5	0x00000004	00 11 22 33 44 55 66 77	Send
6	0x00000005	00 11 22 33 44 55 66 77	Send
7	0x00000006	00 11 22 33 44 55 66 77	Send
8	0x00000007	00 11 22 33 44 55 66 77	Send
9	0x00000008	00 11 22 33 44 55 66 77	Send
10	0x00000009	00 11 22 33 44 55 66 77	Send

COM Open OK

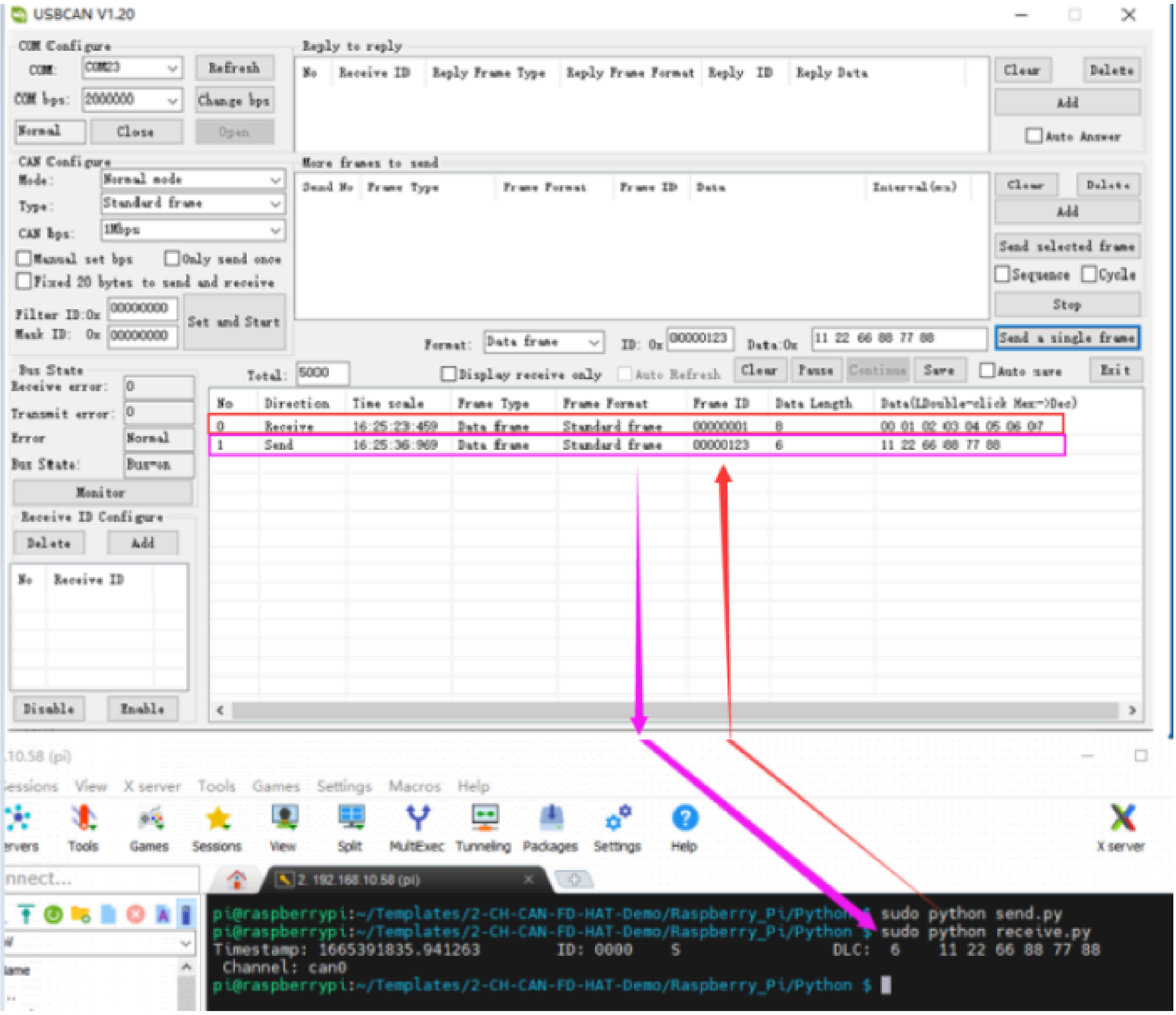
(/wiki/File:CAN-A-2.12-TR.png)

2-CH CAN FD HAT Connection Test

- Connect USB-CAN-A and the 2-CH CAN FD HAT (https://www.waveshare.com/2-ch-can-fd-hat.htm) with cables. Connect A-->A and B-->B, and connect the USB port of USB-CAN-A to the computer USB port:

- Open USB-CAN debug tool on the computer and its default baud rate is 2M. Open the corresponding port, select the normal mode, the standard frame and the corresponding rate of the demo is 1Mbps (USB-CAN-A supports up to 1M). Set and boot it. Then, you can see the bidirectional data transmission between 2-CH CAN FD HAT (https://www.waveshare.com/2-ch-can-fd-hat.htm) and USB-CAN-A

As shown below:



COM Configure

COM: COM23

COM bps: 2000000

Normal

CAN Configure

Mode: Normal mode

Type: Standard frame

CAN bps: 1Mbps

☐ Manual set bps ☐ Only send once

☐ Fixed 20 bytes to send and receive

Filter ID:0x 00000000

Mask ID: 0x 00000000

Format: Data frame ID: 0x 00000123 Data:0x 11 22 66 88 77 88

Bus State

Receive error: 0

Transmit error: 0

Error: Normal

Bus State: Bus-on

Total: 5000

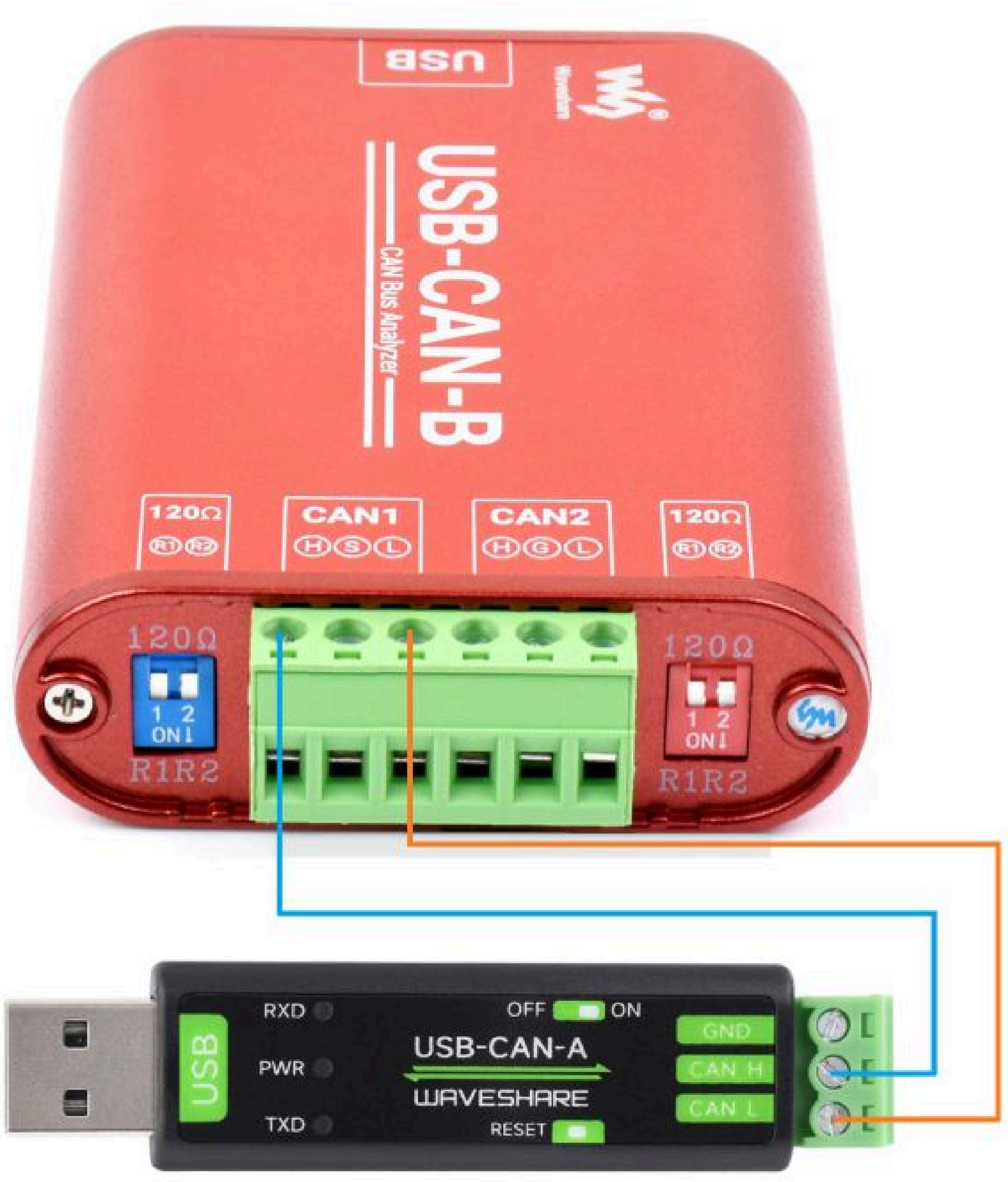
No	Direction	Time scale	Frame Type	Frame Format	Frame ID	Data Length	Data(LDouble-click Hex->Dec)
0	Receive	16:25:23:459	Data frame	Standard frame	00000001	8	00 01 02 03 04 05 06 07
1	Send	16:25:36:969	Data frame	Standard frame	00000123	6	11 22 66 88 77 88

```
pi@raspberrypi:~/Templates/2-CH-CAN-FD-HAT-Demo/Raspberry_Pi/Python $ sudo python send.py
pi@raspberrypi:~/Templates/2-CH-CAN-FD-HAT-Demo/Raspberry_Pi/Python $ sudo python receive.py
Timestamp: 1665391835.941263        ID: 0000    S                  DLC:  6    11 22 66 88 77 88
 Channel: can0
pi@raspberrypi:~/Templates/2-CH-CAN-FD-HAT-Demo/Raspberry_Pi/Python $ 
```

(/wiki/File:800px-USB-CAN-A_RS485-CAN-HAT.png)

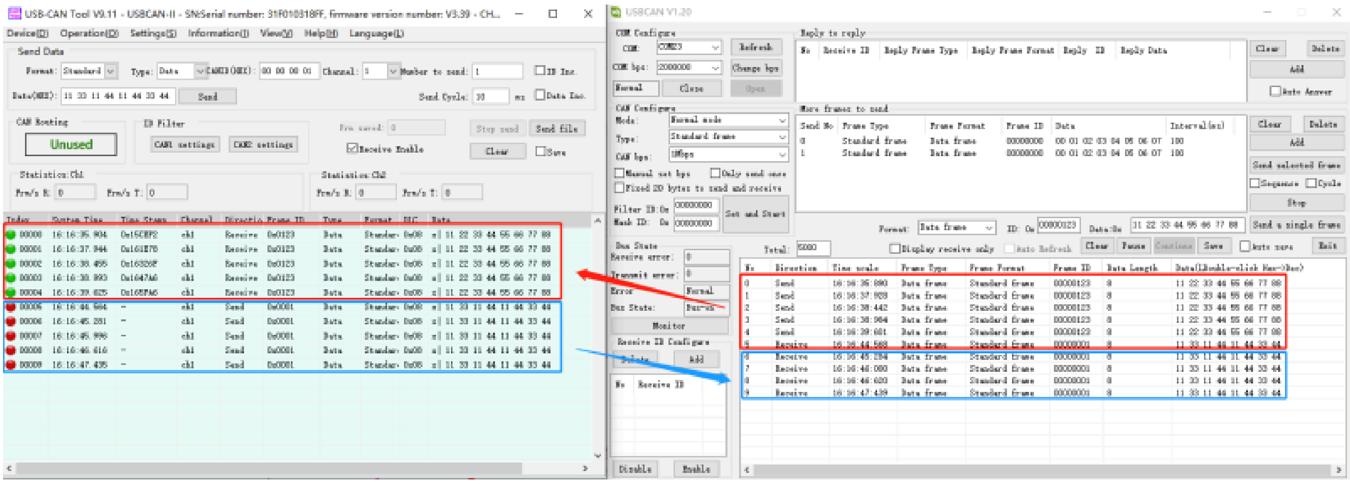
USB-CAN-B Connection Test

- Connect USB-CAN-A and USB-CAN-B (https://www.waveshare.com/usb-can-b.htm) by H-H and L-L with cables. Then, turn on their 120 Ohm resistive switches:



(/wiki/File:USB-CAN-A-B.jpg)

- Enter the USBCAN configuration tool configuration interface -> Confirm the CAN configuration is correct and set it up -> Enter the data information and click Send a single frame; USB-CAN-A sends data and USB-CAN-B receives it, and vice versa, test OK



USB-CAN Tool V9.11 - USBCAN-II - SN:Serial number: 31F010318FF, firmware version number: V3.39 - CH...

Send Data
Format: Standard Type: Data CANID(HEX): 00 00 00 01 Channel: 1 Number to send: 1 □ID Inc.
Data(HEX): 11 33 11 44 11 44 33 44 Send Send Cycle: 10 ms □Data Inc.

CAN Routing: Unused
ID Filter: CAN1 settings CAN2 settings
Fra saved: 0 ☑Receive Enable □Save
Statistics:Ch1 Fra/s R: 0 Fra/s T: 0
Statistics:Ch2 Fra/s R: 0 Fra/s T: 0

Index	System Time	Time Stamp	Channel	Directio	Frame ID	Type	Format	DLC	Data
00000	16:16:35.904	0x15CBF2	ch1	Receive	0x0123	Data	Standar	0x08	x| 11 22 33 44 55 66 77 88
00001	16:16:37.944	0x161E78	ch1	Receive	0x0123	Data	Standar	0x08	x| 11 22 33 44 55 66 77 88
00002	16:16:38.455	0x16326F	ch1	Receive	0x0123	Data	Standar	0x08	x| 11 22 33 44 55 66 77 88
00003	16:16:38.993	0x1647A6	ch1	Receive	0x0123	Data	Standar	0x08	x| 11 22 33 44 55 66 77 88
00004	16:16:39.025	0x165FA6	ch1	Receive	0x0123	Data	Standar	0x08	x| 11 22 33 44 55 66 77 88
00005	16:16:44.564	-	ch1	Send	0x0001	Data	Standar	0x08	x| 11 33 11 44 11 44 33 44
00006	16:16:45.281	-	ch1	Send	0x0001	Data	Standar	0x08	x| 11 33 11 44 11 44 33 44
00007	16:16:45.996	-	ch1	Send	0x0001	Data	Standar	0x08	x| 11 33 11 44 11 44 33 44
00008	16:16:46.616	-	ch1	Send	0x0001	Data	Standar	0x08	x| 11 33 11 44 11 44 33 44
00009	16:16:47.435	-	ch1	Send	0x0001	Data	Standar	0x08	x| 11 33 11 44 11 44 33 44

USBCAN V1.20

COM Configure
COM: COM23 Refresh
COM bps: 2000000 Change bps
Normal Close Open

CAN Configure
Mode: Normal mode
Type: Standard frame
CAN bps: 1Mbps
□Manual set bps □Only send once
□Fixed 20 bytes to send and receive
Filter ID:0x 00000000
Mask ID: 0x 00000000 Set and Start

Bus State
Receive error: 0
Transmit error: 0
Error: Normal
Bus State: Bus-on
Monitor

Receive ID Configure
Delete Add

No	Receive ID

Disable Enable

Reply to reply

No	Receive ID	Reply Frame Type	Reply Frame Format	Reply ID	Reply Data

Clear Delete Add □Auto Answer

More frames to send

Send No	Frame Type	Frame Format	Frame ID	Data	Interval(ms)
0	Standard frame	Data frame	00000000	00 01 02 03 04 05 06 07	100
1	Standard frame	Data frame	00000000	00 01 02 03 04 05 06 07	100

Clear Delete Add Send selected frame □Sequence □Cycle Stop

Format: Data frame ID: 0x 00000123 Data:0x 11 22 33 44 55 66 77 88 Send a single frame

Total: 5000 □Display receive only □Auto Refresh Clear Pause Continue Save □Auto save Exit

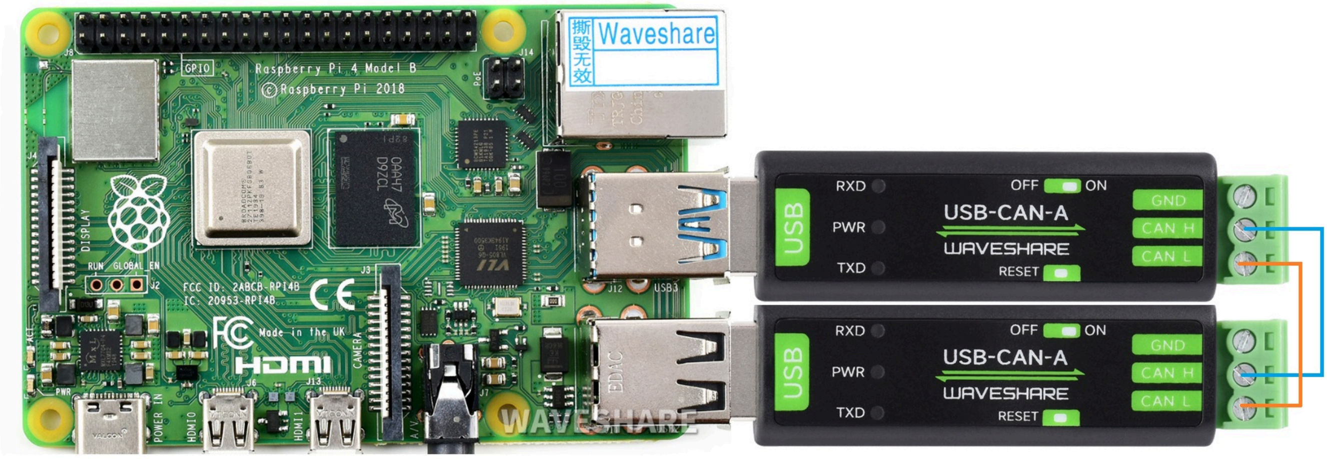
No	Direction	Time scale	Frame Type	Frame Format	Frame ID	Data Length	Data(Double-click Hex->Dec)
0	Send	16:16:35:890	Data frame	Standard frame	00000123	8	11 22 33 44 55 66 77 88
1	Send	16:16:37:928	Data frame	Standard frame	00000123	8	11 22 33 44 55 66 77 88
2	Send	16:16:38:442	Data frame	Standard frame	00000123	8	11 22 33 44 55 66 77 88
3	Send	16:16:38:984	Data frame	Standard frame	00000123	8	11 22 33 44 55 66 77 88
4	Send	16:16:39:601	Data frame	Standard frame	00000123	8	11 22 33 44 55 66 77 88
5	Receive	16:16:44:588	Data frame	Standard frame	00000001	8	11 33 11 44 11 44 33 44
6	Receive	16:16:45:284	Data frame	Standard frame	00000001	8	11 33 11 44 11 44 33 44
7	Receive	16:16:46:000	Data frame	Standard frame	00000001	8	11 33 11 44 11 44 33 44
8	Receive	16:16:46:620	Data frame	Standard frame	00000001	8	11 33 11 44 11 44 33 44
9	Receive	16:16:47:439	Data frame	Standard frame	00000001	8	11 33 11 44 11 44 33 44

(/wiki/File:800px-USB-CAN-A_USB-CAN-B.png)

Working with Linux

Hardware Preparation

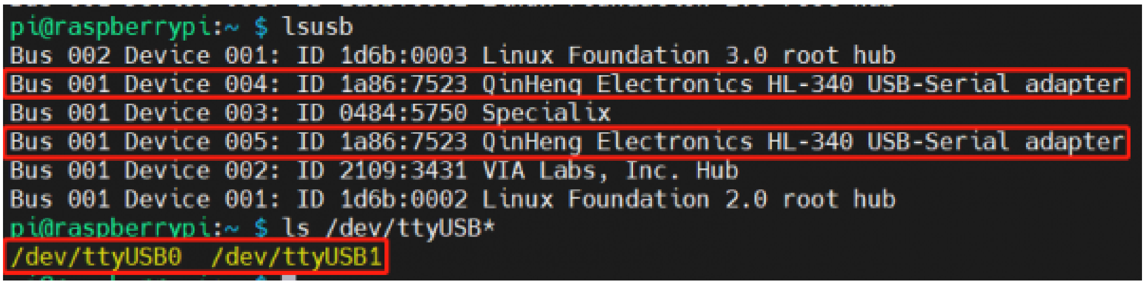
- The L and H of the two USB-CAN-A are connected respectively, turn on the 120 ohm resistive switch of the two (the switch is dialed to the side near the green terminal block). Connect the USB ends to the USB ports of the Raspberry Pi. The hardware connection is shown in the following diagram:



(/wiki/File:2-USB-CAN-A.jpg)

- The following commands query and confirm that the Raspberry Pi recognizes the USB-CAN-A and loads the related driver:

```
lsusb
ls /dev
```



```
pi@raspberrypi:~ $ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 004: ID 1a86:7523 QinHeng Electronics HL-340 USB-Serial adapter
Bus 001 Device 003: ID 0484:5750 Specialix
Bus 001 Device 005: ID 1a86:7523 QinHeng Electronics HL-340 USB-Serial adapter
Bus 001 Device 002: ID 2109:3431 VIA Labs, Inc. Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
pi@raspberrypi:~ $ ls /dev/ttyUSB*
/dev/ttyUSB0  /dev/ttyUSB1
```

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

- For some Linux systems that do not automatically load driver descriptors, you can use the following commands to load the driver:

```
sudo su
sudo modprobe option
sudo sh -c 'echo "1a86 7523" > /sys/bus/usb-serial/drivers/option1/new_id'
```

Demo Test

- Copy the following commands to the terminal of the Raspberry Pi:

```
wget https://files.waveshare.com/wiki/USB-CAN-A/Demo/USB-CAN-A.zip
sudo apt-get install unzip
unzip USB-CAN-A.zip
cd USB-CAN-A
sudo make clean
sudo make
```

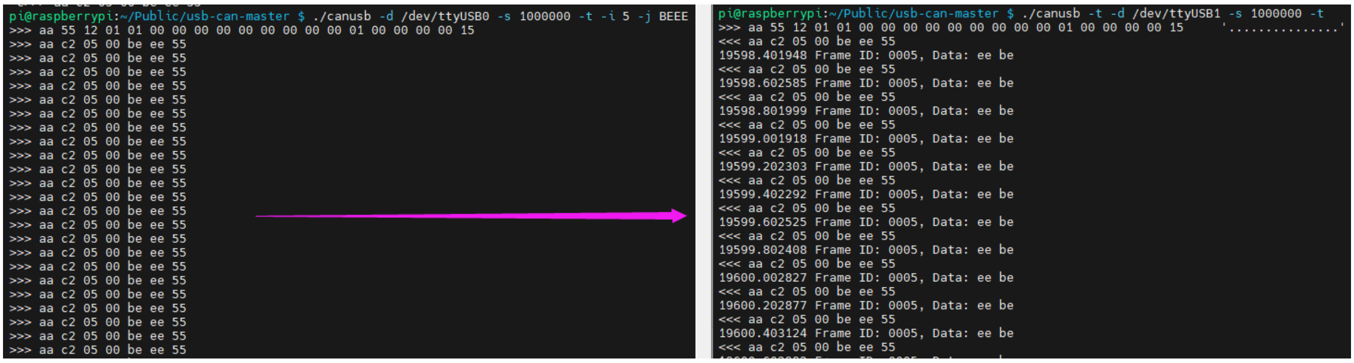
- Enter the following commands in the terminal interface:

```
sudo ./canusb -t -d /dev/ttyUSB0 -s 1000000 -t
```

- Open another terminal interface and run the following sending command:

```
sudo ./canusb -d /dev/ttyUSB1 -s 1000000 -t -i 5 -j BEEE
```

- One USB-CAN-A (ttyUSB0) sends data, and the other USB-CAN-A (ttyUSB1) receives the corresponding data. The effect is as follows:



```
pi@raspberrypi:~/Public/usb-can-master $ ./canusb -d /dev/ttyUSB0 -s 1000000 -t -i 5 -j BEEE
>>> aa 55 12 01 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
>>> aa c2 05 00 be ee 55
```

```
pi@raspberrypi:~/Public/usb-can-master $ ./canusb -t -d /dev/ttyUSB1 -s 1000000 -t
>>> aa 55 12 01 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15     '...............'
<<< aa c2 05 00 be ee 55
19598.401948 Frame ID: 0005, Data: ee be
<<< aa c2 05 00 be ee 55
19598.602585 Frame ID: 0005, Data: ee be
<<< aa c2 05 00 be ee 55
19598.801999 Frame ID: 0005, Data: ee be
<<< aa c2 05 00 be ee 55
19599.001918 Frame ID: 0005, Data: ee be
<<< aa c2 05 00 be ee 55
19599.202303 Frame ID: 0005, Data: ee be
<<< aa c2 05 00 be ee 55
19599.402292 Frame ID: 0005, Data: ee be
<<< aa c2 05 00 be ee 55
19599.602525 Frame ID: 0005, Data: ee be
<<< aa c2 05 00 be ee 55
19599.802408 Frame ID: 0005, Data: ee be
<<< aa c2 05 00 be ee 55
19600.002827 Frame ID: 0005, Data: ee be
<<< aa c2 05 00 be ee 55
19600.202877 Frame ID: 0005, Data: ee be
<<< aa c2 05 00 be ee 55
19600.403124 Frame ID: 0005, Data: ee be
<<< aa c2 05 00 be ee 55
```

(/wiki/File:Usb-can-a_linux10.png)

Resources

Software

- CH341SER driver (https://files.waveshare.com/wiki/common/CH341SER.zip)
- USB-CAN-A_TOOL_1.2 (default 2M serial port baud rate, no need to change) (https://files.waveshare.com/upload/8/8a/USB-CAN-A_TOOL_EN.zip)

- USB-CAN-A_TOOL_2.12 (the latest version) (https://files.waveshare.com/wiki/USB-CAN-A/Tool/USBCANV2.12_English.zip)

Demo

- Linux-C demo (https://files.waveshare.com/wiki/USB-CAN-A/Demo/USB-CAN-A.zip)
- Linux-Python demo (https://files.waveshare.com/wiki/USB-CAN-A/Demo/python/USB-CAN-A-py.zip)

Documents

- Secondary Development—Serial Conversion Definition of CAN Protocol (https://www.waveshare.com/wiki/Secondary_Development_Serial_Conversion_Definition_of_CAN_Protocol)

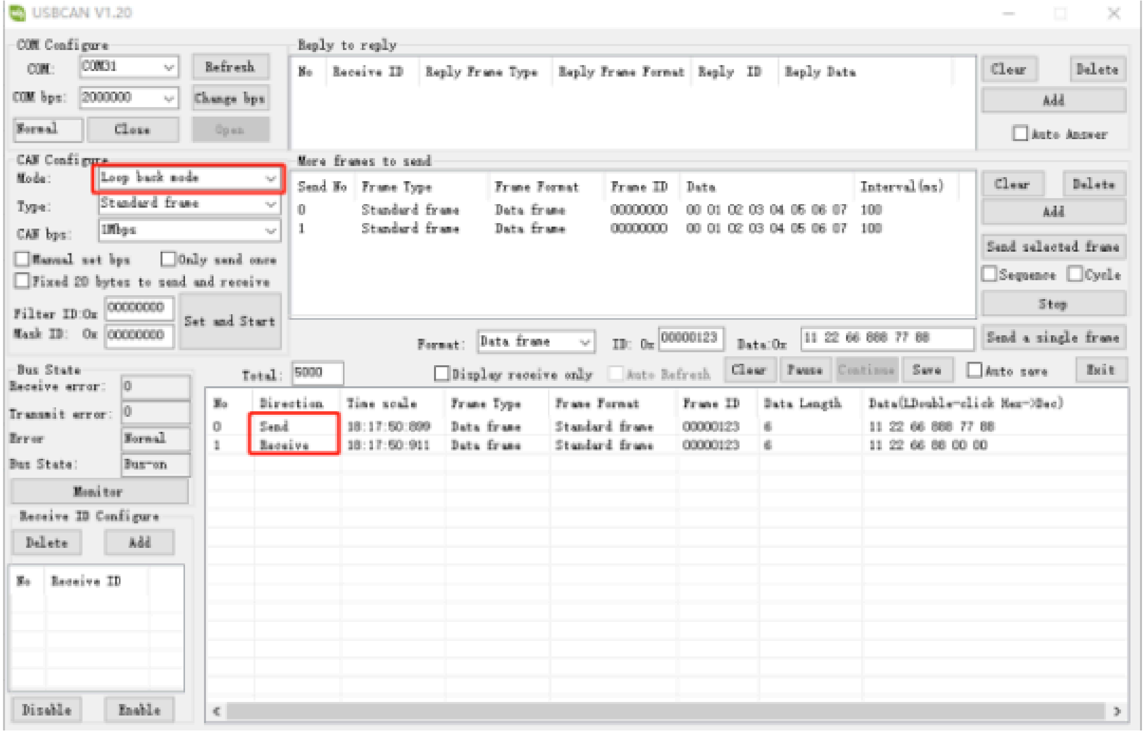
FAQ

Software Settings

Question: I only have a USB-CAN-A on hand, how can I tell that its function is OK?

Answer:

- Please check the baud rate first. If the TX and RX indicators blink once when it is powered, the default setting is right (the serial port baudrate is 2M).
- Please set the module to loopback mode, 2M serial port baud rate (the baud rate is fixed on the computer USB side), CAN rate customizable (set to the same rate as the external device, for example, if the CAN motor is 1M, set it to 1M), set and start. If sending data receives the same data, it indicates that the function is OK:



USBCAN V1.20

COM Configure
COM: COM31
COM bps: 2000000
Normal

CAN Configure
Mode: Loop back mode
Type: Standard frame
CAN bps: 1Mbps
□ Manual set bps □ Only send once
□ Fixed 20 bytes to send and receive
Filter ID:0x 00000000
Mask ID: 0x 00000000

Bus State
Receive error: 0
Transmit error: 0
Error Normal
Bus State: Bus-on

Receive ID Configure

No	Receive ID

Reply to reply

No	Receive ID	Reply Frame Type	Reply Frame Format	Reply ID	Reply Data

□ Auto Answer

More frames to send

Send No	Frame Type	Frame Format	Frame ID	Data	Interval(ms)
0	Standard frame	Data frame	00000000	00 01 02 03 04 05 06 07	100
1	Standard frame	Data frame	00000000	00 01 02 03 04 05 06 07	100

□ Sequence □ Cycle

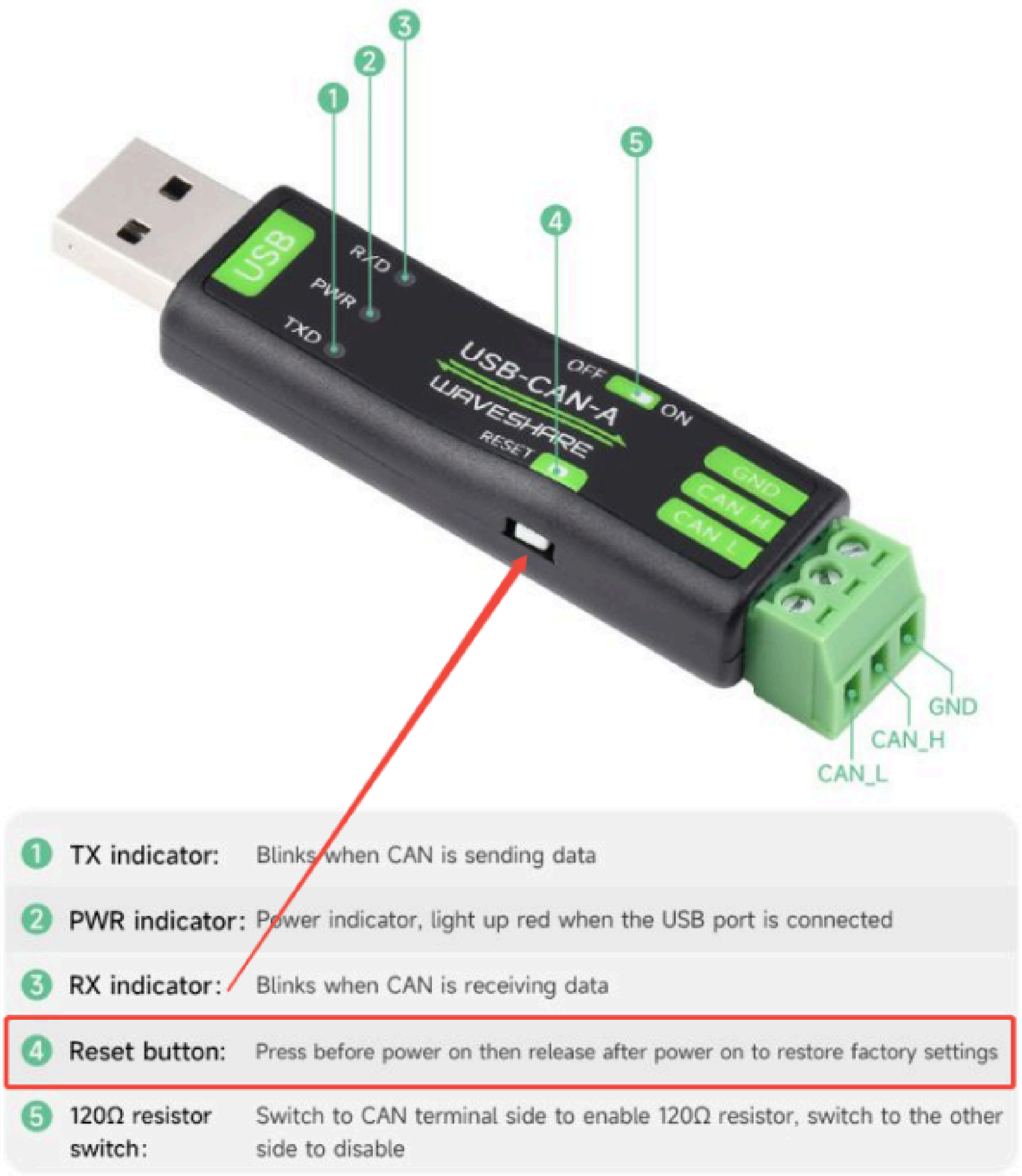
Format: Data frame ID: 0x 00000123 Data:0x 11 22 66 888 77 88

Total: 5000 □ Display receive only □ Auto Refresh □ Auto save

No	Direction	Time scale	Frame Type	Frame Format	Frame ID	Data Length	Data(LDouble-click Hex->Dec)
0	Send	18:17:50:899	Data frame	Standard frame	00000123	6	11 22 66 888 77 88
1	Receive	18:17:50:911	Data frame	Standard frame	00000123	6	11 22 66 88 00 00

(/wiki/File:650px-USB-CAN-A_FAQ02.png)

- If the above method can not work, you can restore the module to the default factory setting and test:



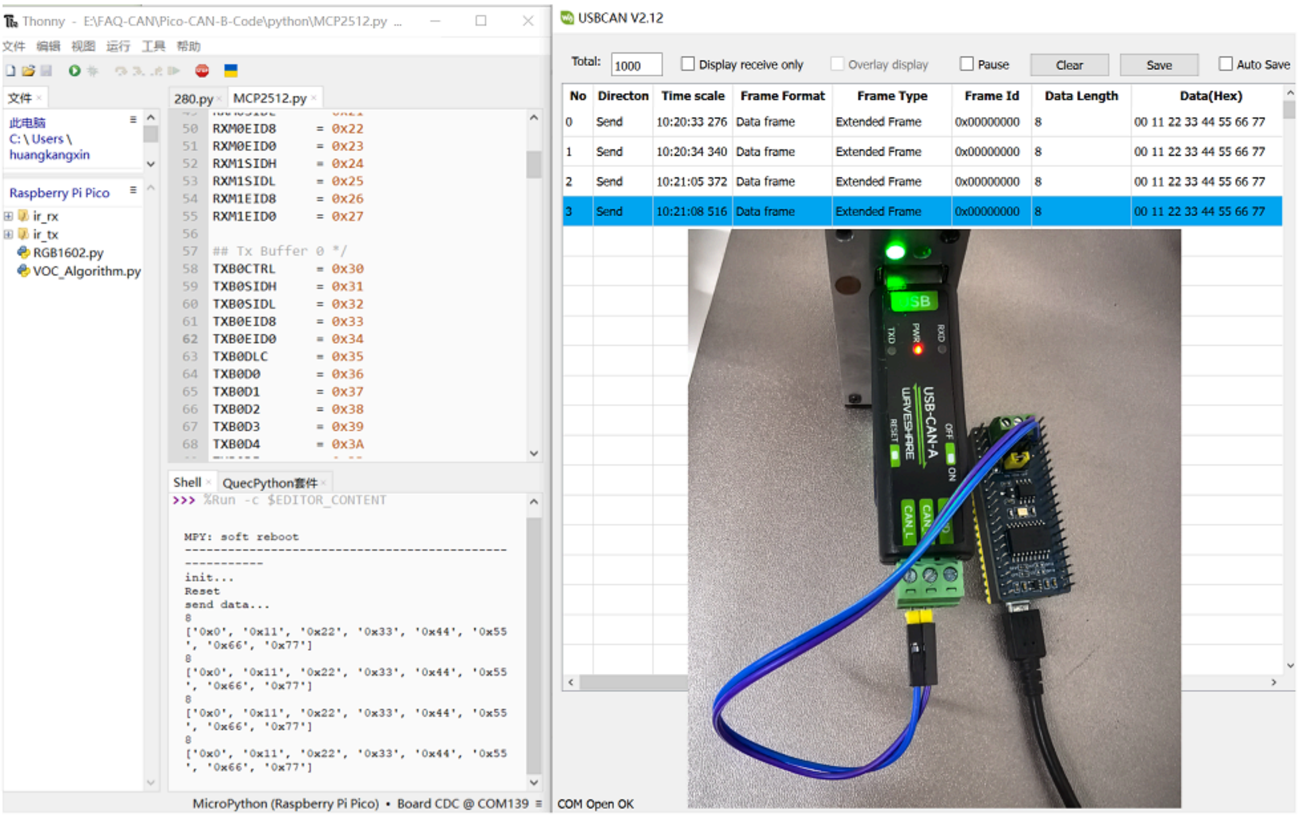
1	TX indicator:	Blinks when CAN is sending data
2	PWR indicator:	Power indicator, light up red when the USB port is connected
3	RX indicator:	Blinks when CAN is receiving data
4	Reset button:	Press before power on then release after power on to restore factory settings
5	120Ω resistor switch:	Switch to CAN terminal side to enable 120Ω resistor, switch to the other side to disable

(/wiki/File:USB-CAN-A-details-inter.png)

Question: There is no problem with the loop test, after confirming that the module is fine, how to test the external CAN device?

Answer:

In CAN communication, parameters such as CAN rate, ID, and data frame format for different devices (motors, mainboards with onboard CAN) may differ. Therefore, when applying in actual scenarios, it is necessary to strictly refer to the specifications of the corresponding CAN devices for confirmation and configuration. As an example, consider a CAN motor with a default CAN rate of 1Mbps, ID of 141 (hexadecimal), and data frame of 88 00 00 00 00 00 77 88 (8 bytes). If you need to achieve communication through the CAN-Tool tool, follow these steps: first, set the CAN rate to 1Mbps in CAN-Tool; second, configure the receive/send ID to 141; finally, input the example data 88 00 00 00 00 00 77 88 in the data frame editing area. After completing the configuration, you can proceed with the communication test. If the actual device parameters do not match the default values, adjust the configuration according to the specifications to ensure normal communication.



```
RXM0EID8        = 0x22
RXM0EID0        = 0x23
RXM1SIDH        = 0x24
RXM1SIDL        = 0x25
RXM1EID8        = 0x26
RXM1EID0        = 0x27

## Tx Buffer 0 */
TXB0CTRL        = 0x30
TXB0SIDH        = 0x31
TXB0SIDL        = 0x32
TXB0EID8        = 0x33
TXB0EID0        = 0x34
TXB0DLC         = 0x35
TXB0D0          = 0x36
TXB0D1          = 0x37
TXB0D2          = 0x38
TXB0D3          = 0x39
TXB0D4          = 0x3A
```

```
>>> %Run -c $EDITOR_CONTENT

 MPY: soft reboot
 ----------------------------------------------------------
 -----------
 init...
 Reset
 send data...
 8
 ['0x0', '0x11', '0x22', '0x33', '0x44', '0x55
 ', '0x66', '0x77']
 8
 ['0x0', '0x11', '0x22', '0x33', '0x44', '0x55
 ', '0x66', '0x77']
 8
 ['0x0', '0x11', '0x22', '0x33', '0x44', '0x55
 ', '0x66', '0x77']
 8
 ['0x0', '0x11', '0x22', '0x33', '0x44', '0x55
 ', '0x66', '0x77']
```

Total: 1000 □ Display receive only □ Overlay display □ Pause Clear Save □ Auto Save

No	Directon	Time scale	Frame Format	Frame Type	Frame Id	Data Length	Data(Hex)
0	Send	10:20:33 276	Data frame	Extended Frame	0x00000000	8	00 11 22 33 44 55 66 77
1	Send	10:20:34 340	Data frame	Extended Frame	0x00000000	8	00 11 22 33 44 55 66 77
2	Send	10:21:05 372	Data frame	Extended Frame	0x00000000	8	00 11 22 33 44 55 66 77
3	Send	10:21:08 516	Data frame	Extended Frame	0x00000000	8	00 11 22 33 44 55 66 77

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

Question: What do the parameters of the Linux commands represent?

Answer:

```
sudo ./canusb -d /dev/ttyUSB1 -s 1000000 -t -i 5 -j BEEE
```

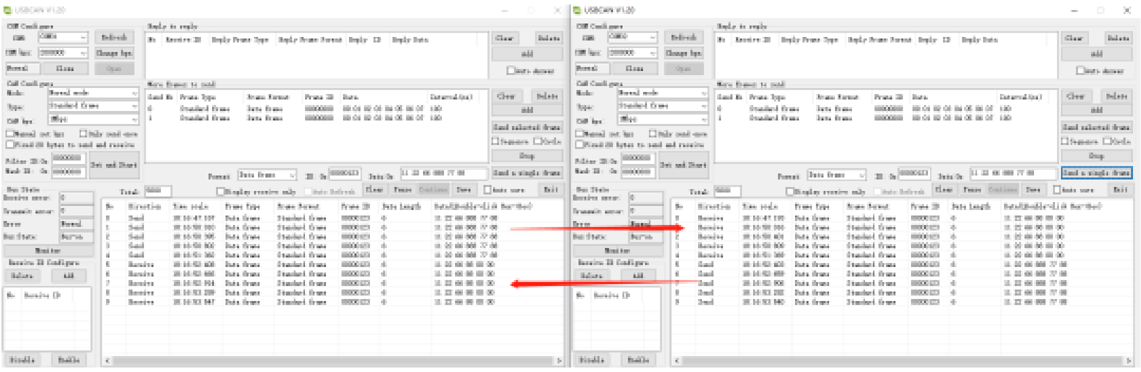
The following is a description of some of the parameters:

```
  -t: Print TTY/serial traffic debug information on stderr.
  -d: The device use TTY device.
  -s: SPEED sets the CAN speed in bps.
  -b: BAUDRATE sets the TTY/serial baud rate (default: 2000000).
  -i: ID uses ID (specified as a hexadecimal string) to inject.
  -j: DATA CAN DATA to be injected (specified as a hexadecimal string).
  -n: COUNT terminates after a COUNT frame (default: infinite).
  -g: MS injects sleep gap in MS milliseconds (default: 200 milliseconds).
  -m: MODE injects load mode (0 = random, 1 = incremental, 2 = fixed).
```

Question: How can I test two USB-CAN-A data transmission and reception?

Answer:

The sender's frame ID should match the receiver's filter ID; both parties must have matching CAN bus rates, and the default baud rate for the serial port is 2 Mbps.



(/wiki/File:USB-CAN-A_FAQ01.png)

Question: Why does the data lose packets?

Answer:

- Data is sent too fast, the first data has not been successfully sent, and the next one is sent, which will also lead to packet loss; When the data is sent, there happens to be data on the bus, which will lead to packet loss. It is recommended to send data packets at intervals of a certain period of time (recommended >1s) or add appropriate delays to give other devices enough time for packet processing. In cases where the module performance is limited, the rate has an upper limit, and there is a large amount of data that cannot be reduced, the frequency needs to be lowered
- Additionally, it is also related to the distance of communication, the cable, the amount of communication data, and the parameter configuration of external devices. For example, under the same conditions, the shorter the cable, the higher the baud rate that can be achieved

Question: Which protocols does USB-CAN-A support?

Answer:

Currently, it supports CAN 2.0A and CAN 2.0B, which are the most basic and common CAN standards. It supports a maximum data transfer rate of 1 Mbps.

Question: Why is the USB-CAN-A serial port baud rate set to 2M?

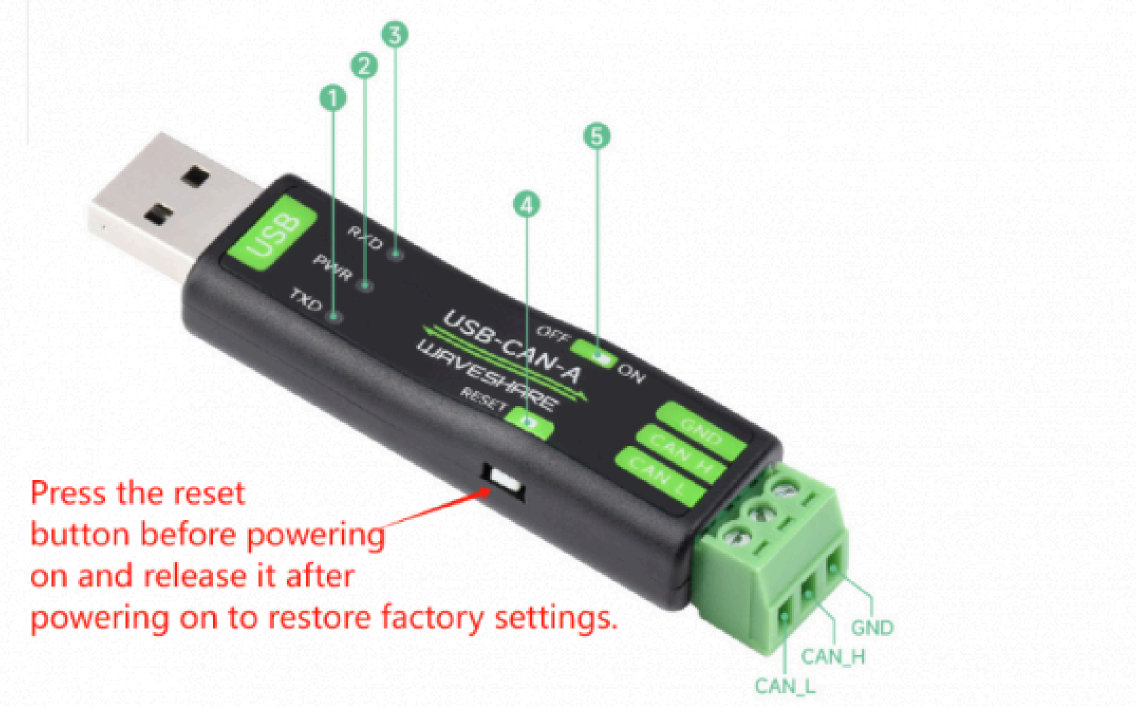
Answer:

This is a USBN to serial port and then to CAN solution, the 2M serial port baud rate can be compatible with most CAN rates, generally the default 2M baud rate is enough, change the CAN rate to the rate corresponding to the external CAN device (up to 1M).

Question: Why wasn't the baud rate change successful?

Answer:

It's not recommended to alter the baud rate. In 99% of cases, 2M is universally applicable. If you belong to the 1% that needs alteration, please choose the default 2M to change. Since you select the default 2M to change, selecting this should take effect. If it still doesn't work, reset to the default 2M before making further changes.



(/wiki/File:USB-CAN-A-faq.png)

Question: Does the USB-CAN-A have message buffering? If so, how many messages can it buffer?

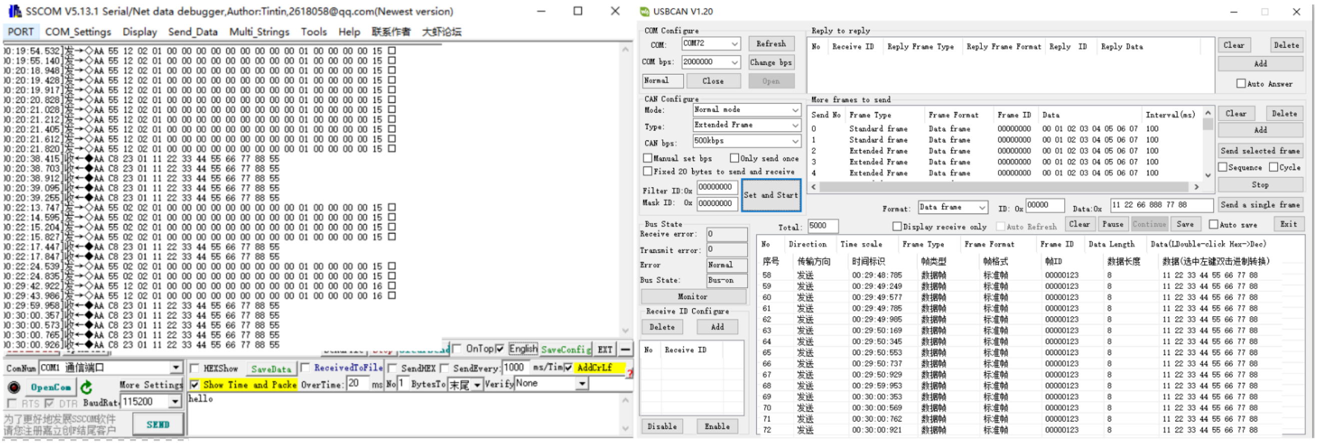
Answer:

Yes, it does. Data is placed in the buffer to prevent errors. If data were forwarded immediately upon reception, errors could occur when new data is received. Hence, buffering is required. Internally, it can buffer up to 20 messages. The converter operates on a mostly real-time conversion basis.

Question: Can you use a different CAN tool as the host computer?

Answer:

- Different CAN configuration tools, the underlying instructions are different, please use our supporting host computer tool for better support
- You can use the CAN-TOOL SDK source code file (https://files.waveshare.com/wiki/USB-CAN-A/Demo/USBCANDemo.zip) of the upper computer tool that integrates corresponding configuration commands here for secondary development of upper computer tools
- Another way is to use the serial port debugging assistant tool to send underlying commands. Note that you should select Fixed 20 Bytes to send and receive as shown in the image:



```
0:19:54.532]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:19:55.140]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:20:18.948]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:20:19.428]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:20:19.917]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:20:20.828]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:20:21.028]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:20:21.212]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:20:21.405]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:20:21.612]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:20:21.820]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:20:38.415]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
0:20:38.703]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
0:20:38.912]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
0:20:39.095]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
0:20:39.255]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
0:22:13.747]发→◇AA 55 02 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:22:14.595]发→◇AA 55 02 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:22:15.204]发→◇AA 55 02 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:22:15.827]发→◇AA 55 02 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:22:17.447]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
0:22:17.847]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
0:22:24.539]发→◇AA 55 02 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:22:24.835]发→◇AA 55 02 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 15 □
0:29:42.922]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 16 □
0:29:43.986]发→◇AA 55 12 02 01 00 00 00 00 00 00 00 00 00 01 00 00 00 00 16 □
0:29:59.958]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
0:30:00.357]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
0:30:00.573]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
0:30:00.765]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
0:30:00.926]收←◆AA C8 23 01 11 22 33 44 55 66 77 88 55
```

□ OnTop ☑ English

ComNum: COM1 通信端口

□ RTS ☑ DTR BaudRate: 115200

□ HEXShow □ ReceivedToFile □ SendHEX □ SendEvery: 1000 ms/Tim ☑ AddCrLf

☑ Show Time and Packe OverTime: 20 ms No 1 BytesTo 末尾 Verify None

hello

COM Configure

COM: COM72

COM bps: 2000000

Normal

CAN Configure

Mode: Normal mode

Type: Extended Frame

CAN bps: 500kbps

□ Manual set bps □ Only send once

□ Fixed 20 bytes to send and receive

Filter ID:0x 00000000

Mask ID: 0x 00000000

Bus State

Receive error: 0

Transmit error: 0

Error: Normal

Bus State: Bus-on

Receive ID Configure

No	Receive ID

Reply to reply

No	Receive ID	Reply Frame Type	Reply Frame Format	Reply ID	Reply Data

□ Auto Answer

More frames to send

Send No	Frame Type	Frame Format	Frame ID	Data	Interval(ms)
0	Standard frame	Data frame	00000000	00 01 02 03 04 05 06 07	100
1	Standard frame	Data frame	00000000	00 01 02 03 04 05 06 07	100
2	Extended Frame	Data frame	00000000	00 01 02 03 04 05 06 07	100
3	Extended Frame	Data frame	00000000	00 01 02 03 04 05 06 07	100
4	Extended Frame	Data frame	00000000	00 01 02 03 04 05 06 07	100

□ Sequence □ Cycle

Format: Data frame ID: 0x 00000 Data:0x 11 22 66 888 77 88

Total: 5000 □ Display receive only □ Auto Refresh □ Auto save

No	Direction	Time scale	Frame Type	Frame Format	Frame ID	Data Length	Data(LDouble-click Hex->Dec)
序号	传输方向	时间标识	帧类型	帧格式	帧ID	数据长度	数据(选中左键双击进制转换)
58	发送	00:29:48:785	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
59	发送	00:29:49:249	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
60	发送	00:29:49:577	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
61	发送	00:29:49:785	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
62	发送	00:29:49:985	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
63	发送	00:29:50:169	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
64	发送	00:29:50:345	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
65	发送	00:29:50:553	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
66	发送	00:29:50:737	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
67	发送	00:29:50:929	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
68	发送	00:29:59:953	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
69	发送	00:30:00:353	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
70	发送	00:30:00:569	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
71	发送	00:30:00:762	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88
72	发送	00:30:00:921	数据帧	标准帧	00000123	8	11 22 33 44 55 66 77 88

(/wiki/File:USB-CAN-A-faq2.png)

Question: How to confirm the baud rate?

Answer:

Flash 1 time, the baud rate of the corresponding serial port is 2000000bps Flash 2 time, the baud rate of the corresponding serial port is 1228800bps Flash 3 time, the baud rate of the corresponding serial port is 115200bps Flash 4 time, the baud rate of the corresponding serial port is 38400bps Flash 5 time, the baud rate of the corresponding serial port is 19200bps Flash 6 time, the baud rate of the corresponding serial port is 9600bps

Question: Can USB-CAN-A receive CAN messages from vehicles?

Answer:

Vehicles meeting the following conditions can be supported:

- Having a standard CAN interface, supporting CAN 2.0 A/B protocol.

- As vehicle interfaces may vary and the protocol can be complex, USB-CAN-A might not be optimally utilized. When communicating, it may be necessary to remove the 120-ohm jumper cap on the USB-CAN-A device. Apart from this consideration, communication is similar to normal devices. As long as the baud rate is consistent, data can be sent and received normally.
- If industrial-grade or vehicle-specific SAE J1939 protocol is required, please use USB-CAN-B (https://www.waveshare.com/usb-can-b.htm) instead.

Question: Can USB-CAN-A function as a node for data transmission and reception?

Answer:

Yes, it can function as a node on the CAN bus for data transmission and reception. By connecting to a PC or other host system through a USB interface, it uses its integrated CAN controller to communicate with the CAN bus network. This setup allows the host system to access the CAN network via the USB interface, enabling data transmission and reception.

Question: Which protocols does USB-CAN-A support?

Answer:

Currently, it supports CAN 2.0A and CAN 2.0B, which are the most basic and common CAN standards. It supports a maximum data transfer rate of 1 Mbps.

Question: Does USB-CAN-A support opencan?

Answer:

Not supported, it is recommended to use USB-CAN-B (https://www.waveshare.com/usb-can-b.htm) that supports opencan

Question: Is there an SDK for the upper computer can tool, and are there corresponding library functions?

Answer:

Please click the link CAN-TOOL SDK source code file (https://files.waveshare.com/wiki/USB-CAN-A/Demo/USBCANDemo.zip)

Question: Does LINUX support SOCKETCAN?

Answer:

Not supported. This is a serial port to CAN conversion, loading USB COM port device (ttyUSB0), cannot load can0 device

- USB-CAN-A -> serial port CAN, cost-effective, simplified system architecture, easy to learn or compatible with old devices
- USB-CAN-B USB-CAN-FD/B-> USB2CAN/PCAN-USB, the benchmark for commercial use, industrial lightweight applications
- Raspberry Pi CAN HAT Series - > socket can, for Raspberry Pi programming learning

Question: Why can't data be received but sending is OK?

Answer:

It might be an ID blocking issue. Please change the blocked ID to 00000000 to accept any ID data

Hardware Interfaces

Question: What is the specification of the green terminal?

Answer:

KF2EDG-3P, spacing 3.81mm

Question: Is the Gnd of USB-CAN-A required to be connected?

Answer:

Optional, CAN and RS485 are differential signals. It is recommended to connect to GND in the following situations:

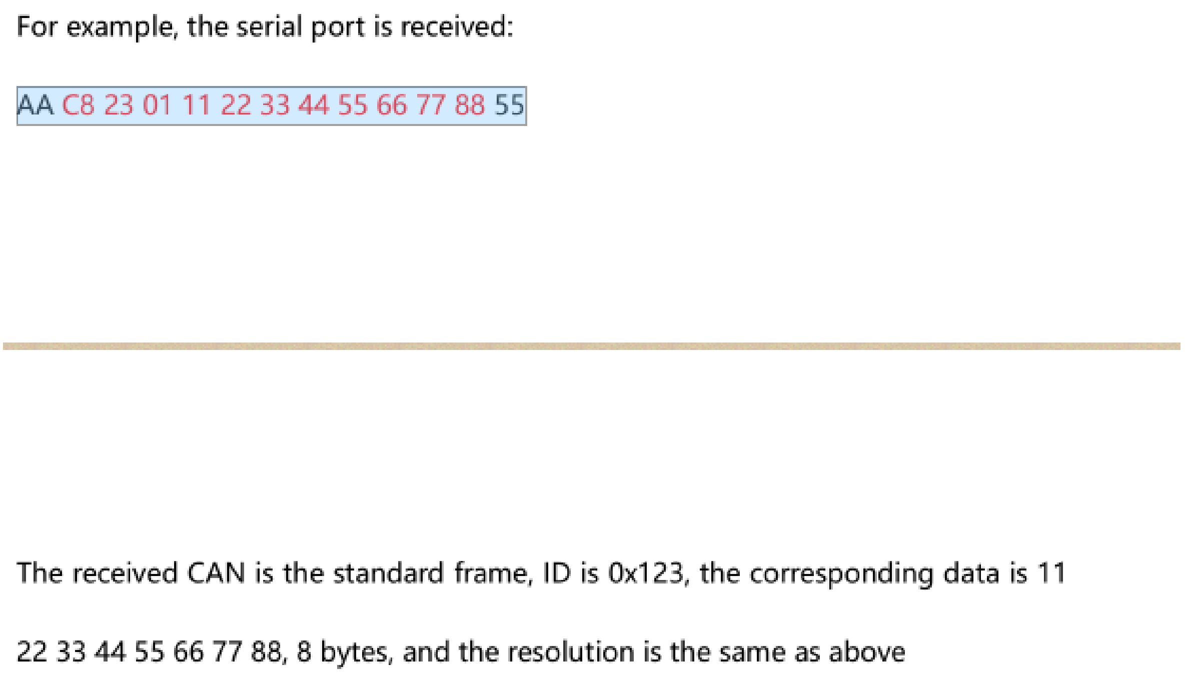
- Long-range communication (>1 meter) —— prevent ground potential difference from interfering with signals.
- Interconnect different power devices——avoid common-mode voltage damage to chips.
- In high-interference environments (such as near industrial areas or motors) — enhance anti-interference capabilities.

Question:I want to know does USB to CAN Adapter Model A has a Hardware interface for python-can or with one is support for it?

Answer:

We haven't developed a demo for python yet. You can try to send it with tools such as our conversion definition and serial port debugging assistant. Basically, anyone who has done programming can quickly understand it;

You can directly send the message inside. Try AA C8 23 01 11 22 33 44 55 66 77 88 55



For example, the serial port is received:

AA C8 23 01 11 22 33 44 55 66 77 88 55

The received CAN is the standard frame, ID is 0x123, the corresponding data is 11 22 33 44 55 66 77 88, 8 bytes, and the resolution is the same as above

(/wiki/File:USB-can-a-faq02.png)

For more details, you can refer to this link (https://files.waveshare.com/wiki/USB-CAN-A/Demo/USB%20(Serial%20port)%20to%20CAN%20protocol%20defines.pdf).

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-A&oldid=105215 (https://www.waveshare.com/w/index.php?title=USB-CAN-A&oldid=105215)"