RS232/485/422 TO CAN

From Waveshare Wiki
Jump to: navigation, search

> Note: If you need to use the CAN to Modbus RTU function, please first check the CAN to Modbus RTU FAQ below, as this feature may not be suitable for your use case.

RS232/485/422 TO CAN



(https://www.waveshare.com/rs232-485-422-to-can.htm)

RS232 RS485 RS422 TO CAN

Overview

Introduction

RS232/485/422 TO CAN is an industrial-grade RS232/485/422 to CAN isolated converter. It features built-in power isolation, ADI magnetic coupling isolation, TVS, and other protection circuits, with an industrial guide rail case. RS232/485/422 TO CAN is easy to operate, with zero-delay automatic send/receive conversion. It possesses characteristics such as fast communication speeds, stability, reliability, and security, making it suitable for various industrial control devices or applications with high communication requirements.

Features

- Adopts SP3232EEN + SP485EEN chips, stable and high-speed communication, with better compatibility.
- Onboard unibody power supply isolation, provides stable isolated voltage and needs no extra power supply for the isolated terminal.
- Onboard unibody digital isolation, allows signal isolation, high reliability, strong anti-interference, and low power consumption.
- Compliant with CAN 2.0B (extended frame) specification, compatible with CAN 2.0A (standard frame) specification; Compliant with ISO 11898-1/2/3 standard.
- CAN baud rate: 10kbps~1000kbps, configurable CAN baud rate.
- High-speed conversion: At a serial port baud rate of 115200 and CAN rate of 250kbps, the CAN sending speed can reach up to 1270 extended frames per second（Close to the theoretical maximum of 1309 extended frames per second）.
- Supports data communication and conversion between RS232/485/422 and CAN interfaces.
- Supports 4 operating modes: transparent conversion, transparent with identifiers conversion, format conversion, and Modbus RTU protocol conversion.

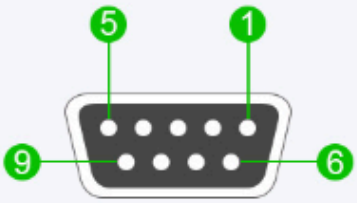
- Onboard TVS (Transient Voltage Suppressor), effectively suppresses surge voltage and transient spike voltage in the circuit, lightningproof & anti-electrostatic.
- Onboard self-recovery fuse and protection diodes, ensure the current/voltage stable outputs, provide over-current/over-voltage proof, and improve shock resistance.
- Onboard 15KV ESD isolation protection and 600W lightningproof & anti-surge protection.
- Onboard 120R terminal resistor on the RS485/RS422 port, configurable by jumper.
- Supports device firmware upgrade via RS232/485/422, more convenient for firmware customization.
- Supports setting operation parameters via configuration software or serial commands, capable of saving settings after powering off.
- Onboard power supply screw terminal, allows 5V~36V DC wide-range input.
- 4 LEDs for indicating the power and transceiver status.
- Industrial grade metal case, supports wall-mount and rail-mount installation, solid and beautiful, easy to install.

Parameters

Product	Industrial isolated RS232/485/422 TO CAN	
Host Interface	CAN	
Device Interface	RS232/485/422	
CAN Interface	Baud Rate	10kbps~1000kbps
	Direction Control	Hardware automatic control
	Resistor	Default 120R
	Interface	Screw terminal
	CAN Interface	H, L, GND
RS232/485/422 Interface	Baud Rate	1200bps ~ 460800bps
	Direction Control	Hardware automatic control
	Resistor	Onboard reserved 120R matching resistor, NC by default, enabled via jumpers
	RS232 Interface	DB9 Interface
	RS485/422 Interface	Screw terminal
	RS485 Interface	A+, B-, GND
	RS422 Interface	TA, TB, RA, RB, GND
	Protection	Provides 600W of lightning, surge, and 15KV static protection (on-board 120R balancing resistor)
	Transmission Mode	Point-to-multipoints (up to 32 nodes, it is recommended to use repeaters for 16 nodes or more)
Button	Press and hold for 1 second	Reset the system
	Press and hold for 5 seconds	Restore Factory Status

Indicator	PWR	Red power indicator, lights up when voltage is detected
	RUN	Operation status indicator, normal operation blinks at 1-second intervals
	COM	CAN blue TX indicator, lights up when the CAN port sends data RS232/485/422 blue RX indicator, lights up when RS232/285/422 port receives data
	CAN	CAN blue RX indicator, lights up when CAN port receives data RS232/485/422 blue TX indicator, lights up when RS232/485/422 port sends data
Environment	Temperature range	-10℃ ~70℃

Interface Introduction



RS232 Female Pinout

RS232 Pinout	
DB9 (PIN)	RS232 PIN
2	Sending data (TXD)
3	Receiving data (RXD)
5	GND
1 , 4 , 6 , 7 , 8 , 9	N/C



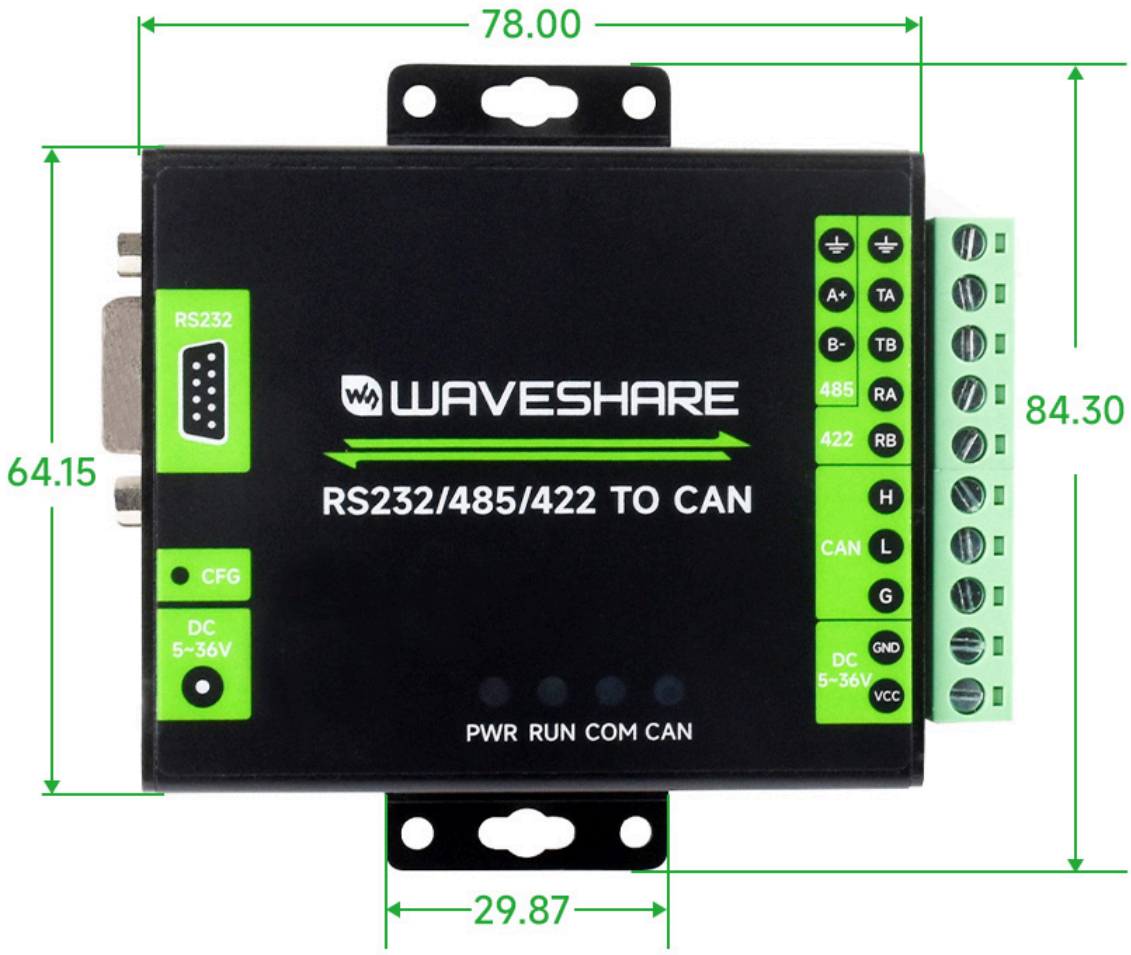
RS485/422 Pinout	
Interface	Definition
PE	RS485/422 signal ground
TA	RS422 send differential signal positive RS485 differential signal positive (A+)
TB	RS422 send differential signal negative RS485 differential signal negative (B-)
RA	RS422 receive differential signal positive
RB	RS422 receive differential signal negative

CAN Pinout	
Pin	Definition
H	CAN_H
L	CAN_L
G	CAN signal ground

DC 5~36V Pinout	
Pin	Definition
GND	Gound
VCC	5-36V power supply

(/wiki/File:RS232_Interface_Introduction.png)

Dimensions


78.00
64.15
84.30
29.87
RS232
CFG
DC
5~36V
WAVESHARE
RS232/485/422 TO CAN
PWR RUN COM CAN
A+
B-
485
TA
TB
RA
422
RB
CAN
H
L
G
DC
5~36V
GND
VCC


WAVESHARE
RS232/485/422 TO CAN
PWR RUN COM CAN
RS232
CFG
DC
5~36V
24.30

Unit: mm

(/wiki/File:RS232-485-422_to_can02.jpg)

Default Factory Settings

Item	Parameter
RS232/485/422	115200,8,N,1
CAN working mode	Transparent conversion, bidirectional
CAN baud rate	250kbps
Type of frame CAN sends	Extended frame
Frame ID CAN sends	0x12345678
CAN filter	Prohibition(Accept all CAN frames)

Device Configuration

Use USB TO 4CH Serial Converter (https://www.waveshare.com/usb-to-4ch-serial-converter.htm) (or USB to RS232, USB to RS485 or USB to RS422) for initial configuration. No configuration is performed, and the factory settings are maintained.

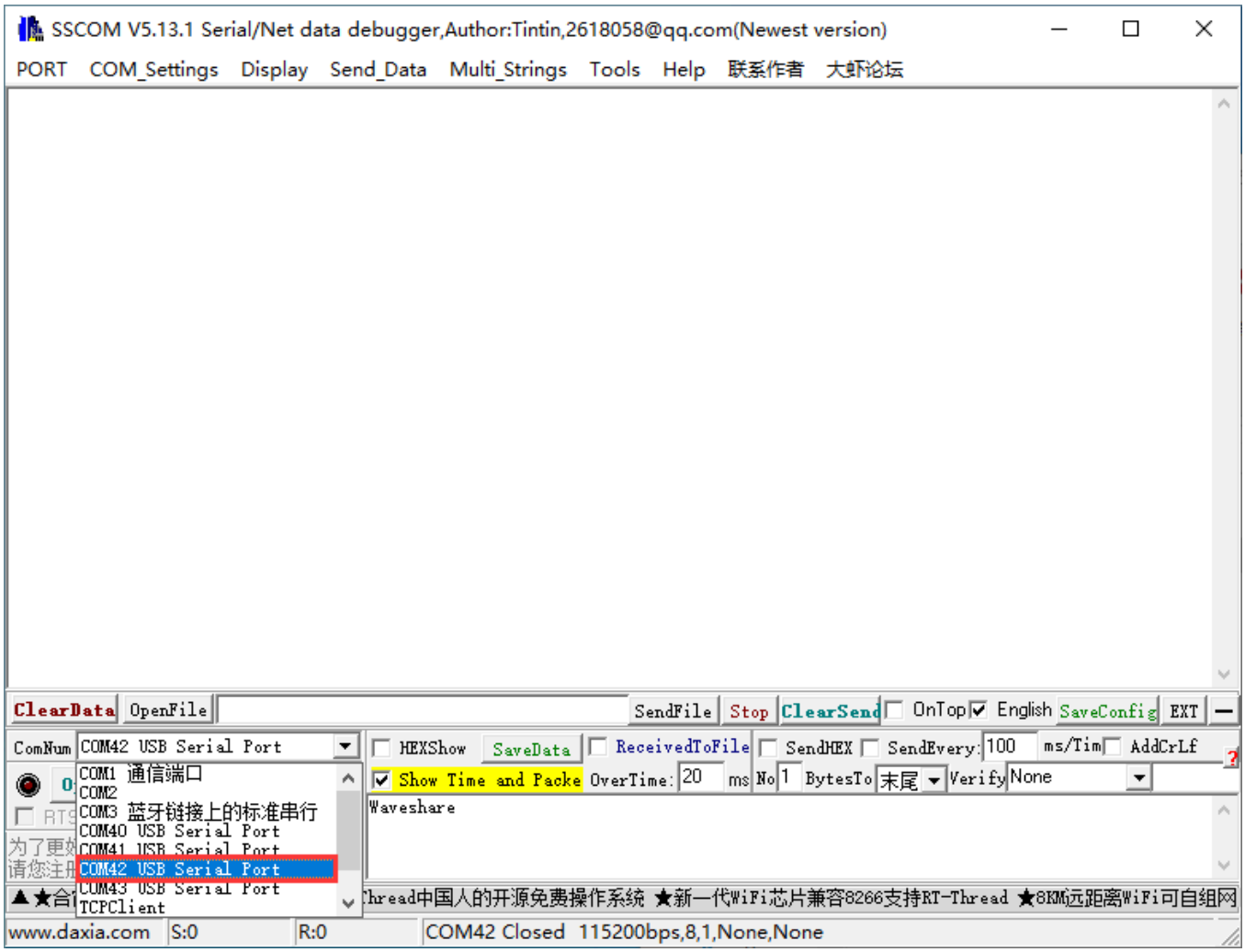
Hardware Connection

- Connect USB to RS232/485/422 device (the following shows RS485 interface connection of USB TO 4CH Serial Converter):

RS232/485/422 TO CAN - RS485	USB TO 4CH Serial Converter - Port C
RS485 - GND	Port C - GND
RS485 - A+	Port C - A+
RS485 - B-	Port C - B-

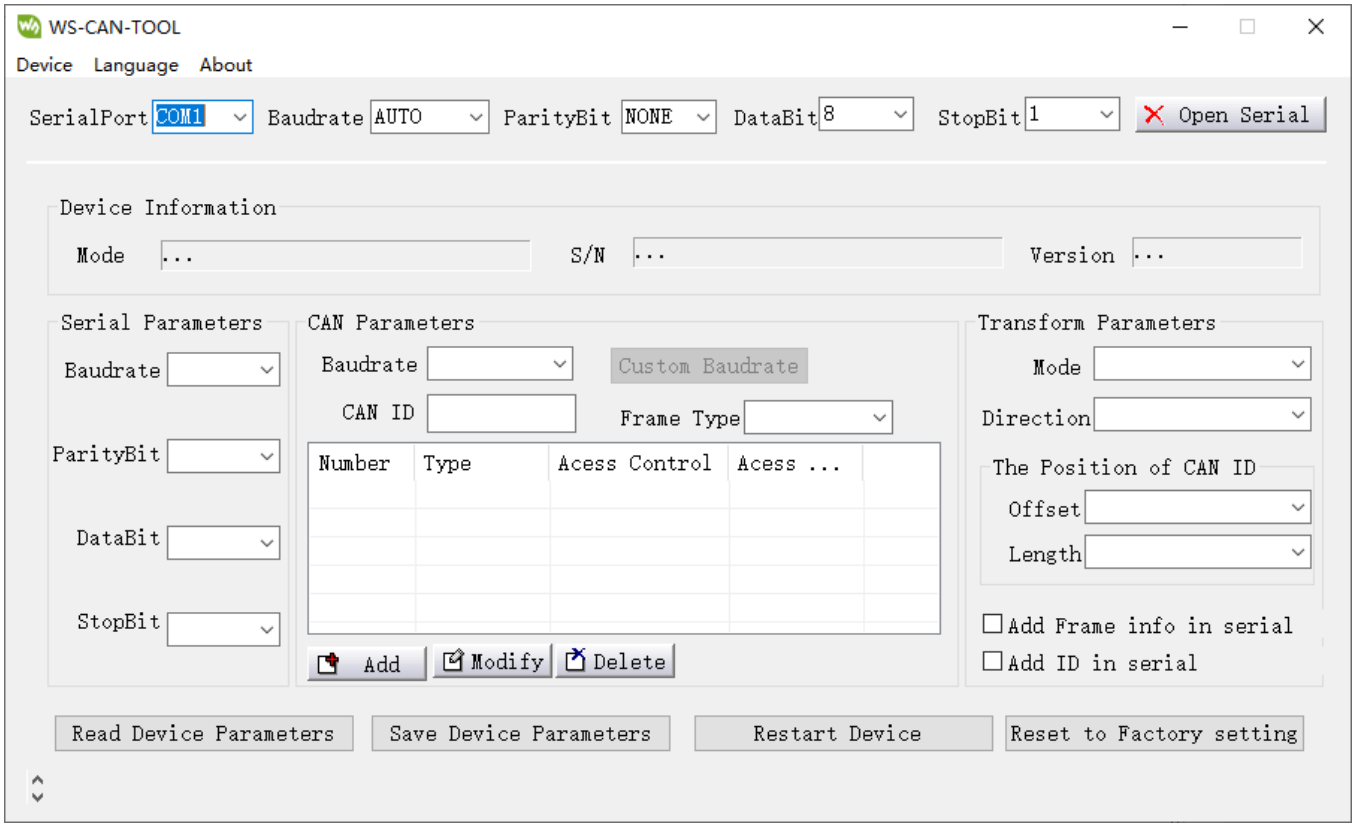
Software Operation

- Open SSCOM, view the COM port used, and perform it with the COM42:



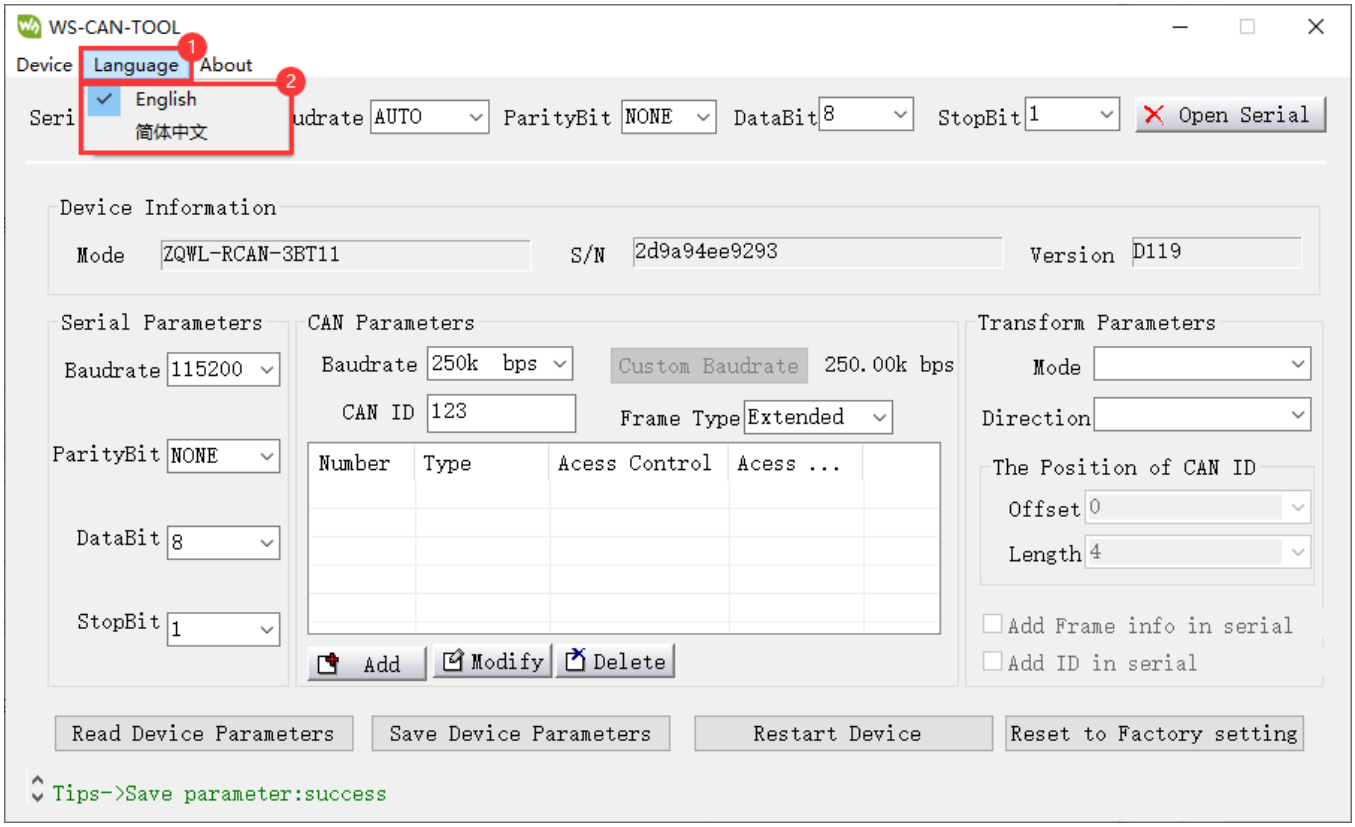
(/wiki/File:RS232-485-422_to_can03.png)

- Open WS-CAN-TOOL (https://files.waveshare.com/wiki/WS-TTL-CAN/tool/WS-CAN-TOOL%20(17).zip)



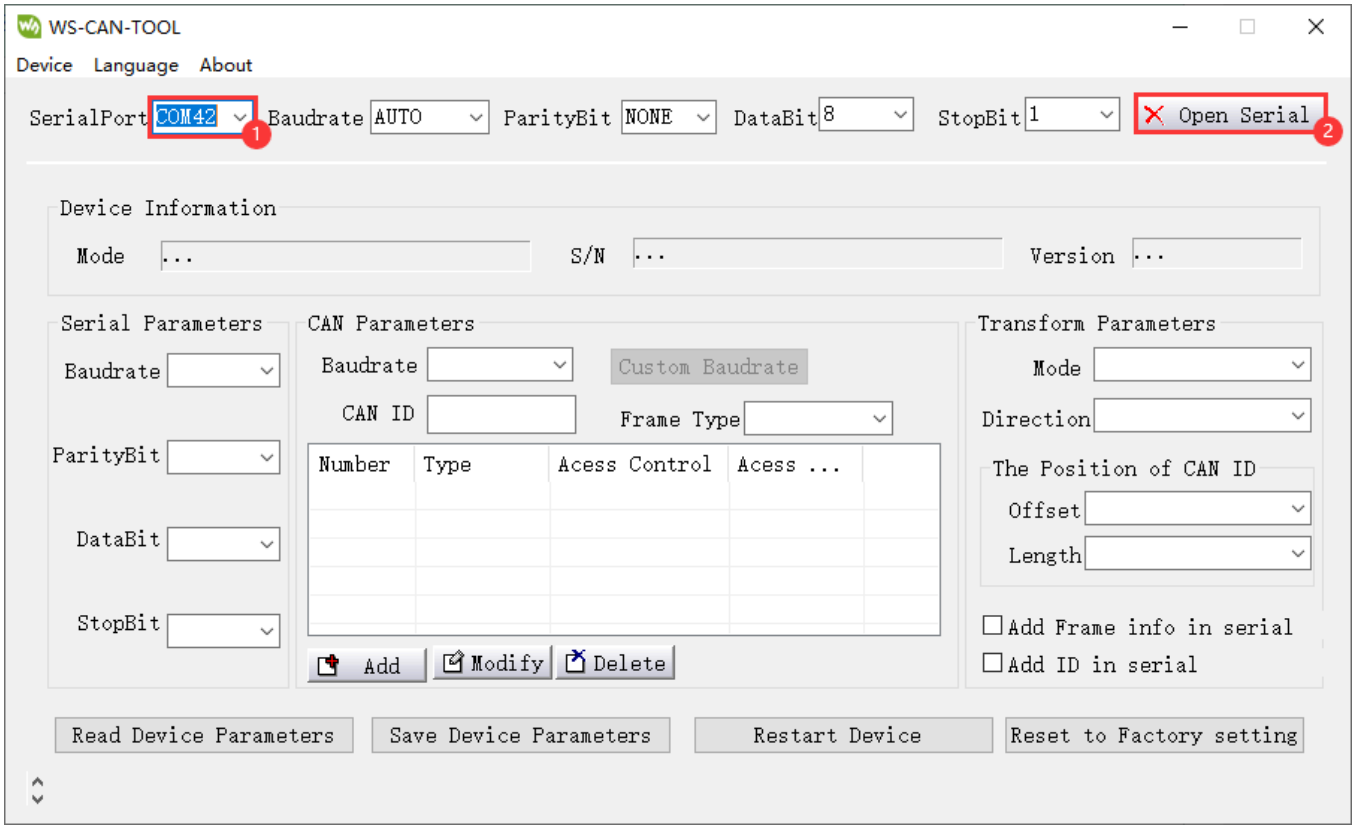
(/wiki/File:RS232-485-422_to_can04.png)

- Select the language:



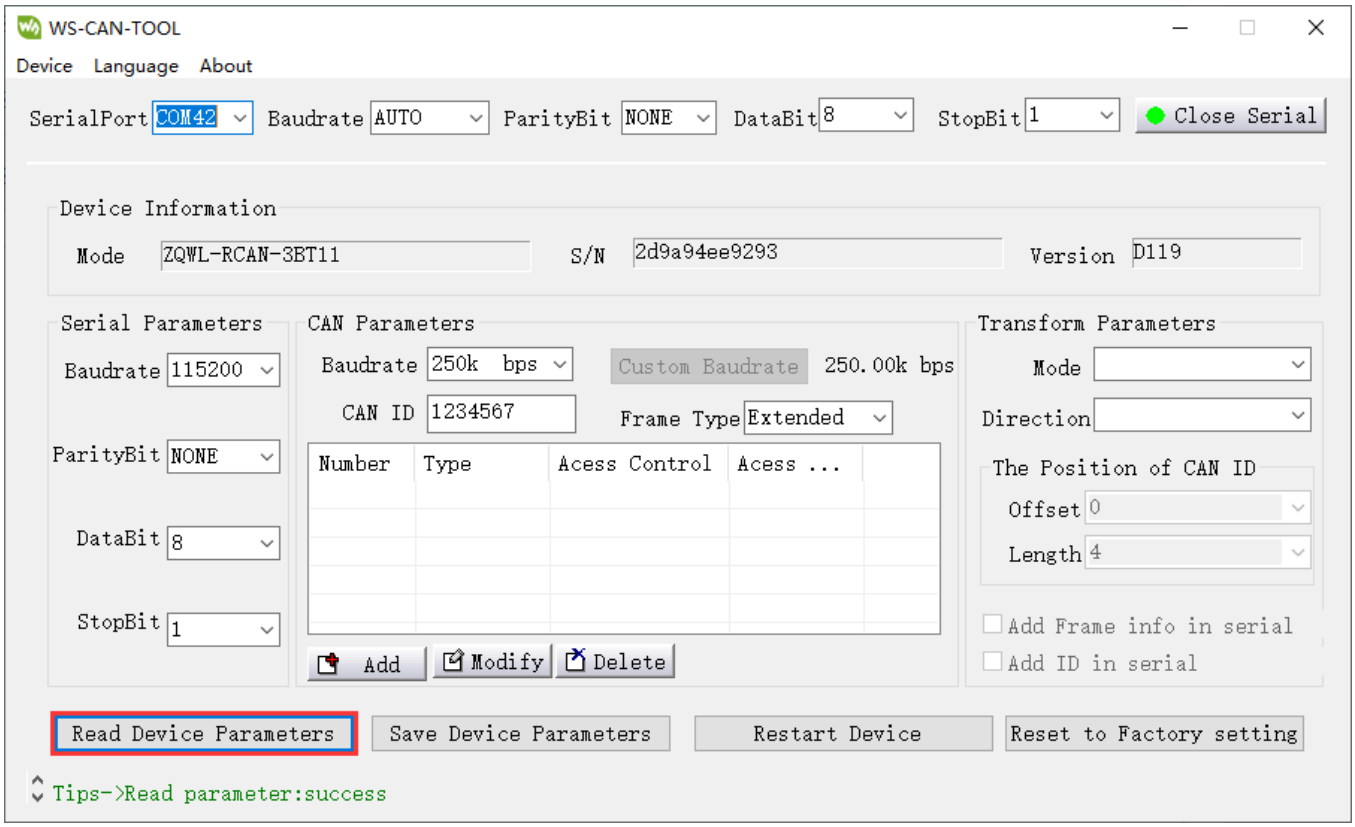
(/wiki/File:RS232-485-422_to_can05.png)

- Select the COM port corresponding to the connected RS232/485/422, and then open the serial port:



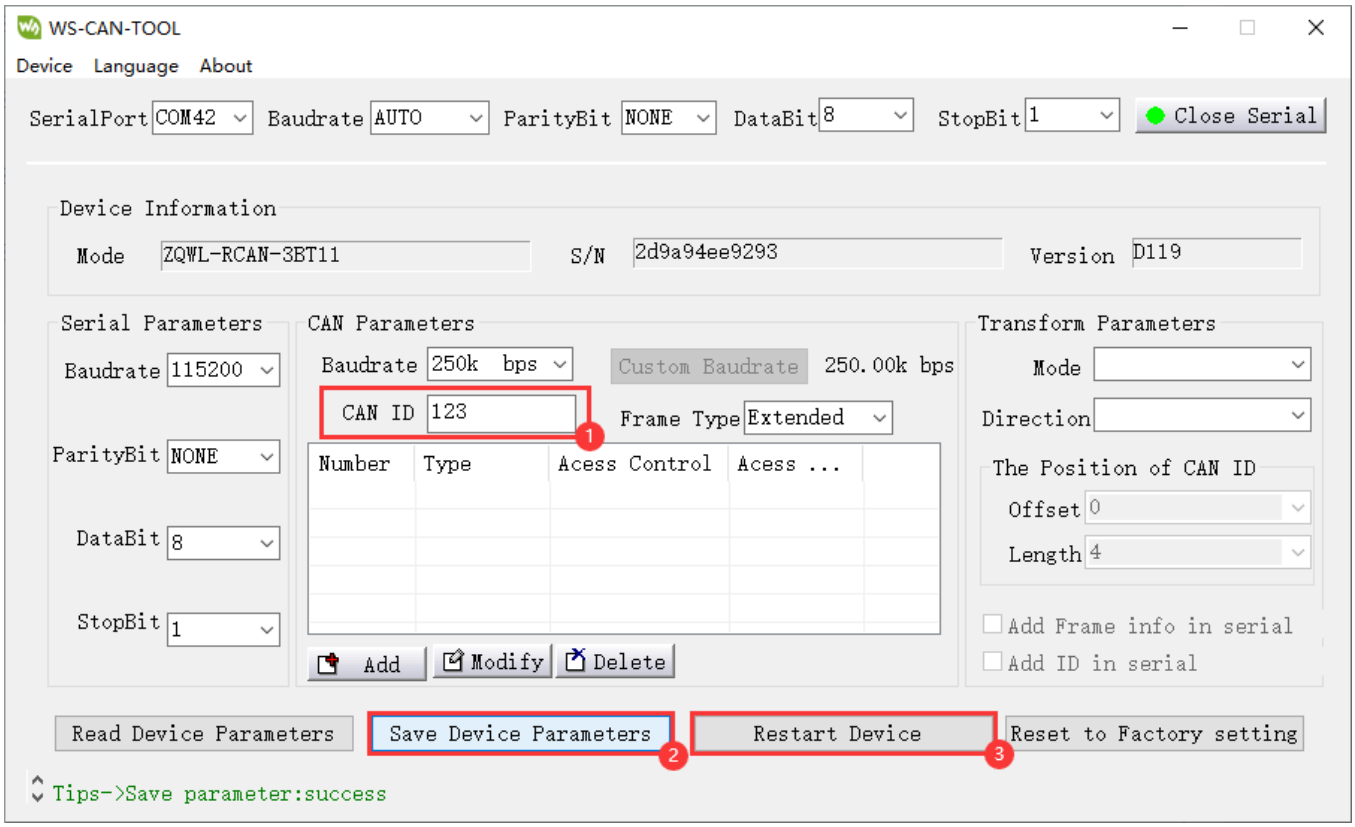
(/wiki/File:RS232-485-422_to_can06.png)

- Click on **Read Device Parameters**.



(/wiki/File:RS232-485-422_to_can07.png)

- If you need to modify the device parameters, you can directly modify it and click on "**Save Device Parameters**", and then click on "**Restart Device**" (for example, modifying CAN ID):



(/wiki/File:RS232-485-422_to_can08.png)

Module Usage Explanation

Transparent Conversion [Expand]

Transparent Conversion With ID [Expand]

Format Conversion [Expand]

Modbus Protocol Conversion [Expand]

Verify the Function

RS232 TO CAN Test

Use USB-CAN-A (https://www.waveshare.com/usb-can-a.htm) and USB TO 4CH Serial Converter (https://www.waveshare.com/usb-to-4ch-serial-converter.htm) to demonstrate.

Hardware Connection

- Connect to the CAN device:

RS232/485/422 TO CAN - CAN	USB-CAN-A
CAN - CAN H	CAN - CAN H
CAN - CAN L	CAN - CAN L

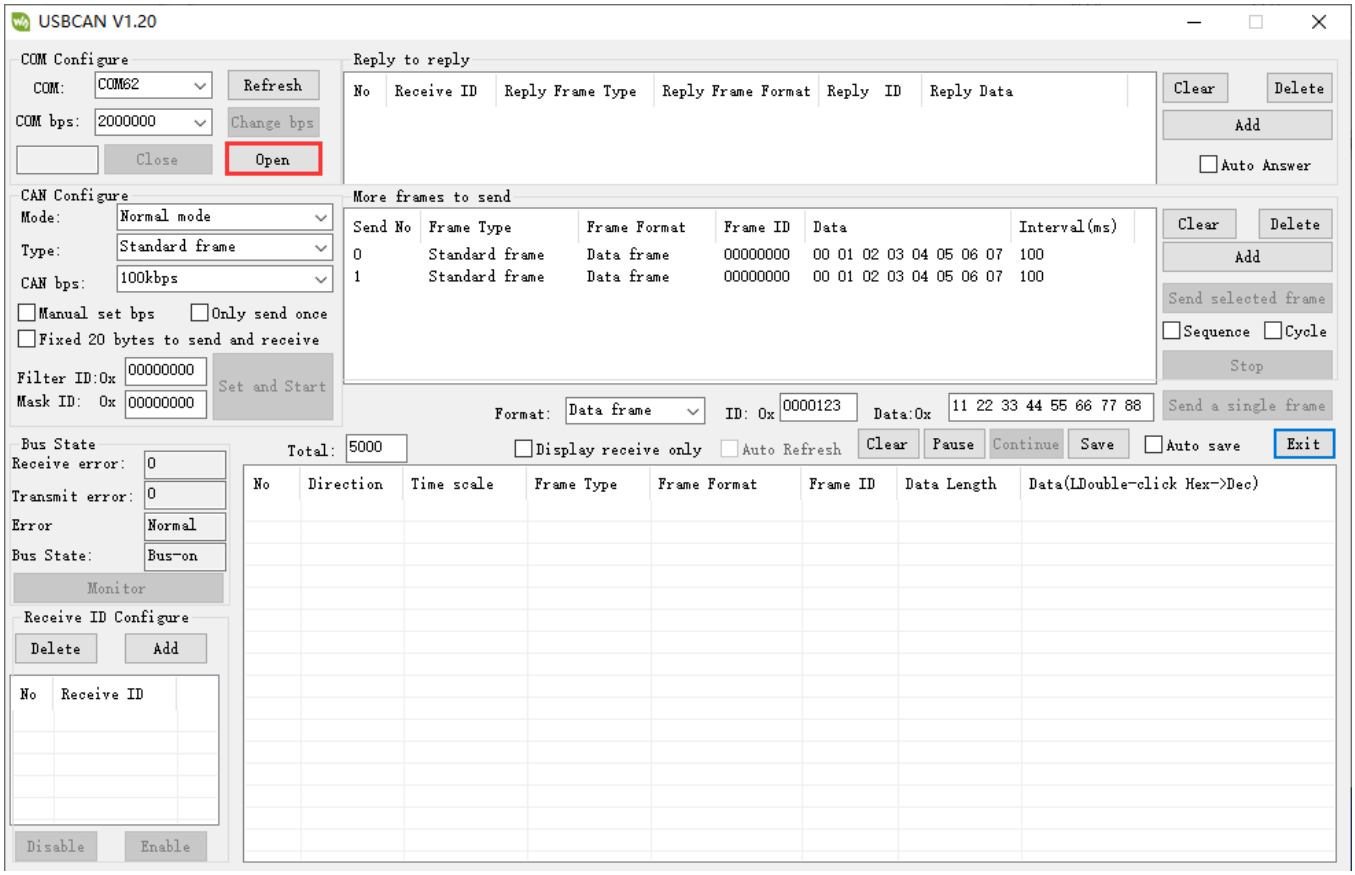
CAN - CAN G	CAN - GND

- Connect to RS232 device

RS232/485/422 TO CAN - RS232	USB TO 4CH Serial Converter - Port D
RS232 - GND	Port D - GND
RS232 - TXD	Port D - RXD
RS232 - RXD	Port D - TXD

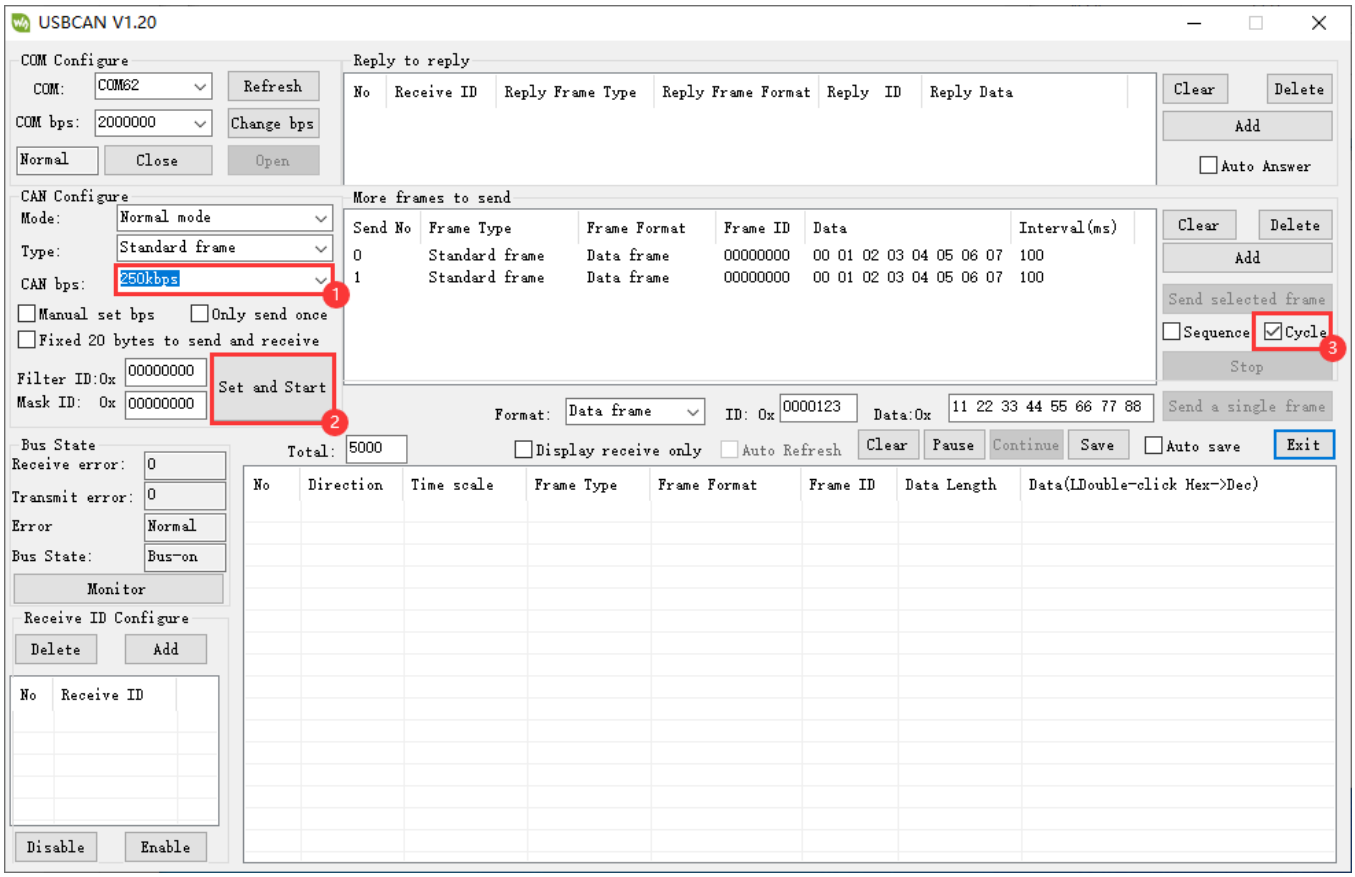
Software Operation

- Open USB-CAN-A-TOOL (https://files.waveshare.com/wiki/RS232-485-422-TO-CAN/USB-CAN-A_TOOL.zip).
- Select the corresponding COM port of the USB-CAN-A, and click on "**Open**".



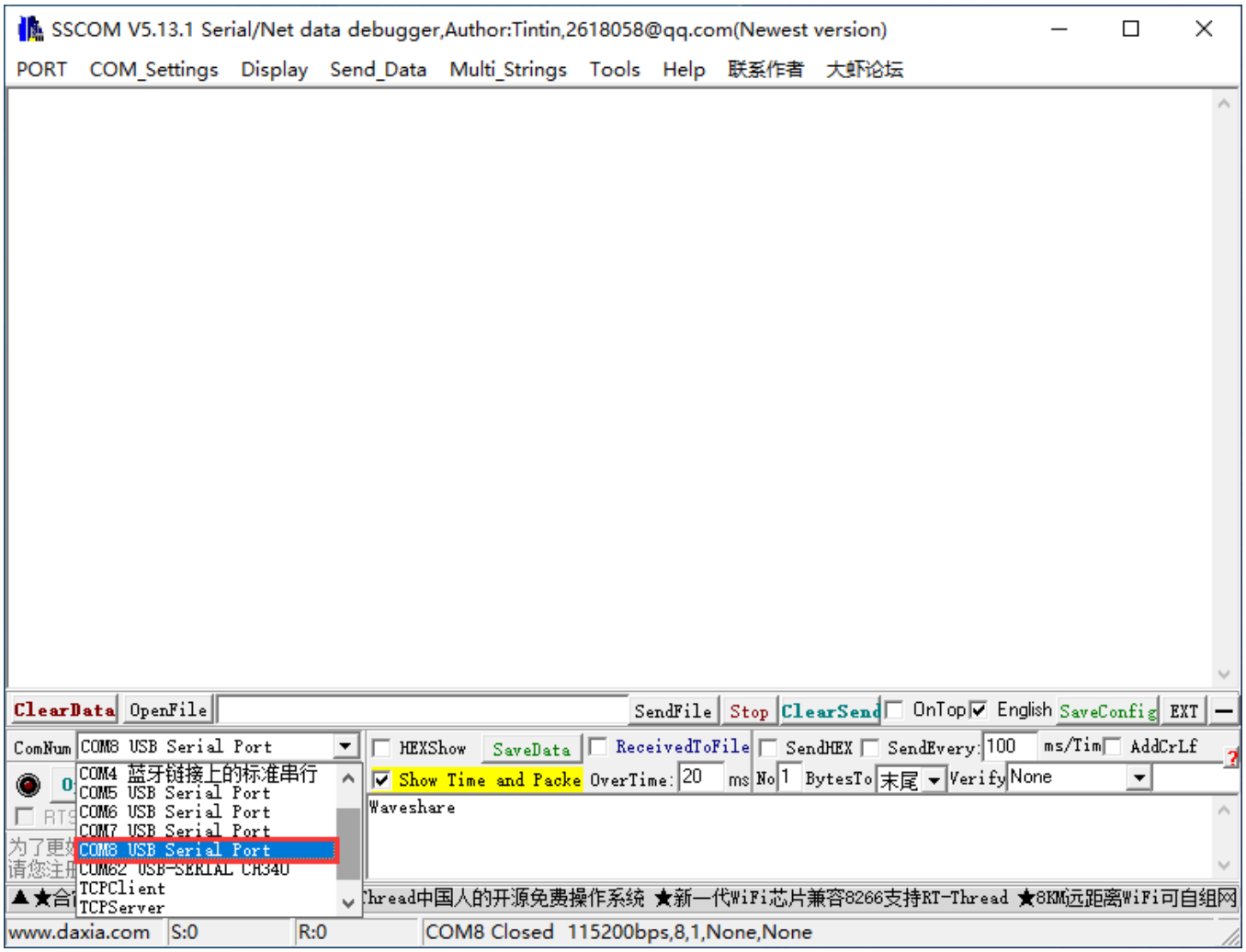
(/wiki/File:RS232_Software_Operation-1.png)

- Select the baud rate of the CAN device as 250kbps, click on "**Set and Start**" , input the data to send, and select "**Cycle**".



(/wiki/File:RS232_Software_Operation-2.png)

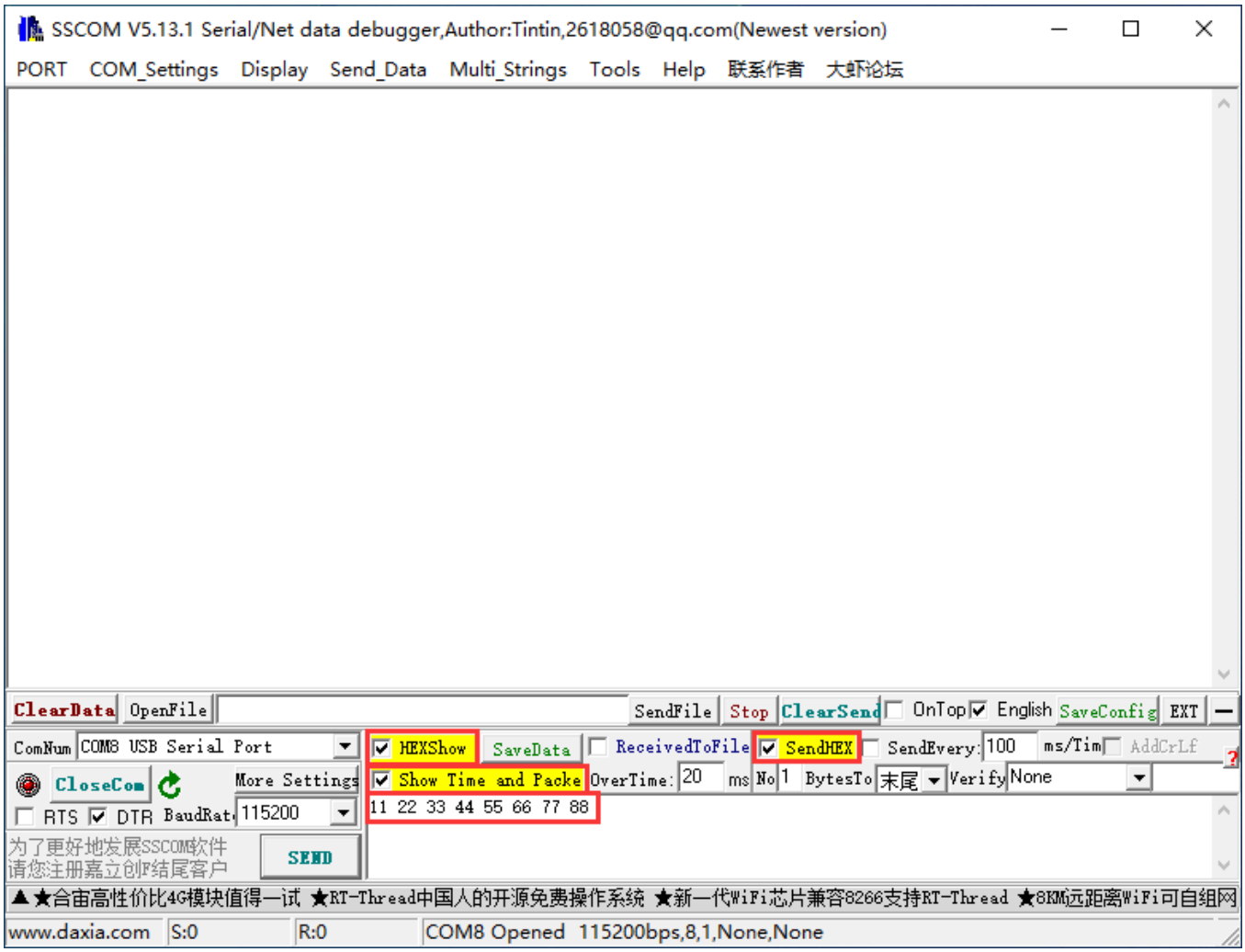
- Open an SSCOM, select the corresponding COM port of the USB TO 4CH Serial Converter's Port D, and open the serial port.



(/wiki/File:RS232_Software_Operation-3.png)

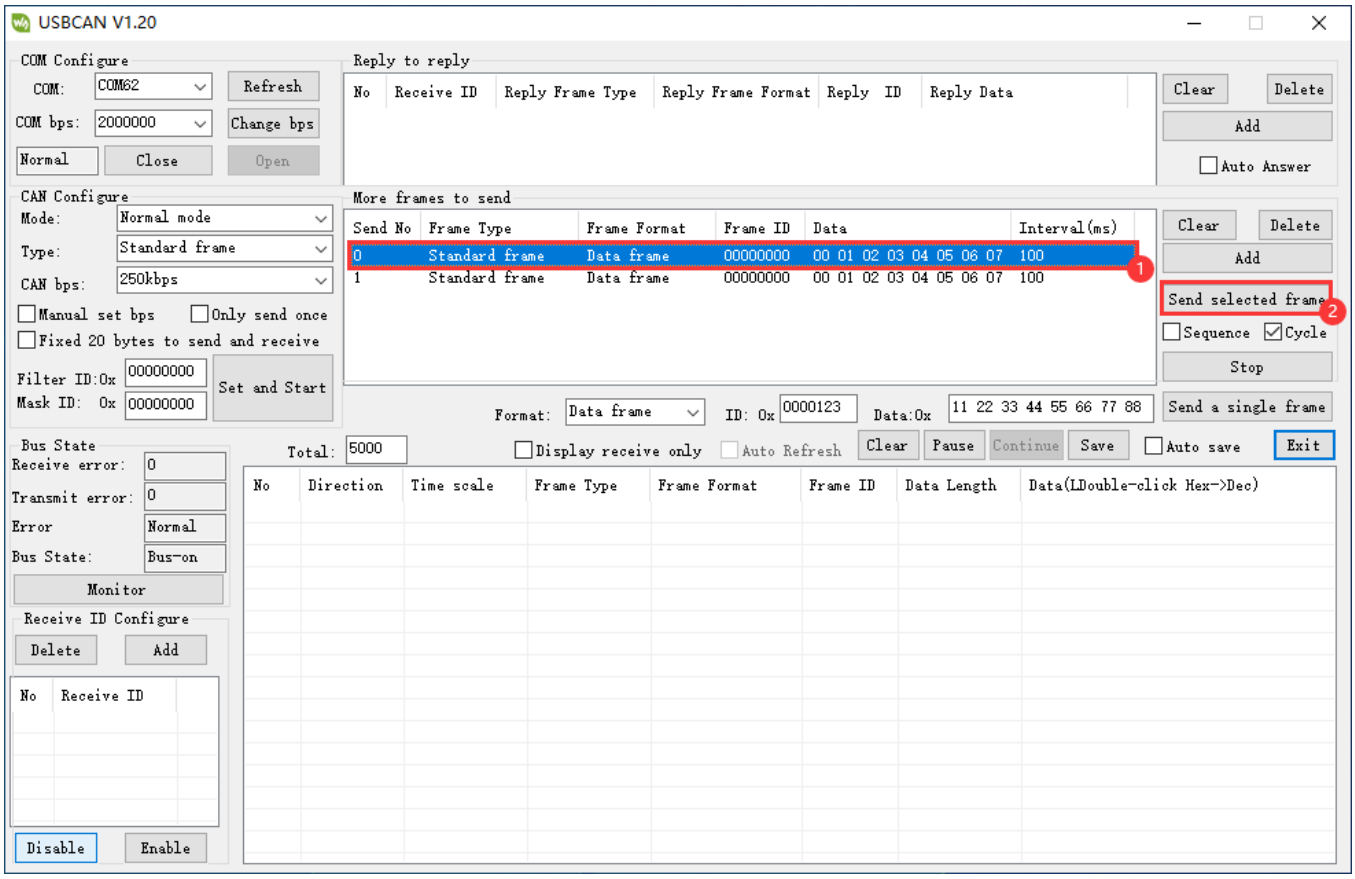
- In the SSCOM software, check "Hexshow", "sendHex" and "Show Time and Package", input the data to be sent.

```
11 22 33 44 55 66 77 88
```



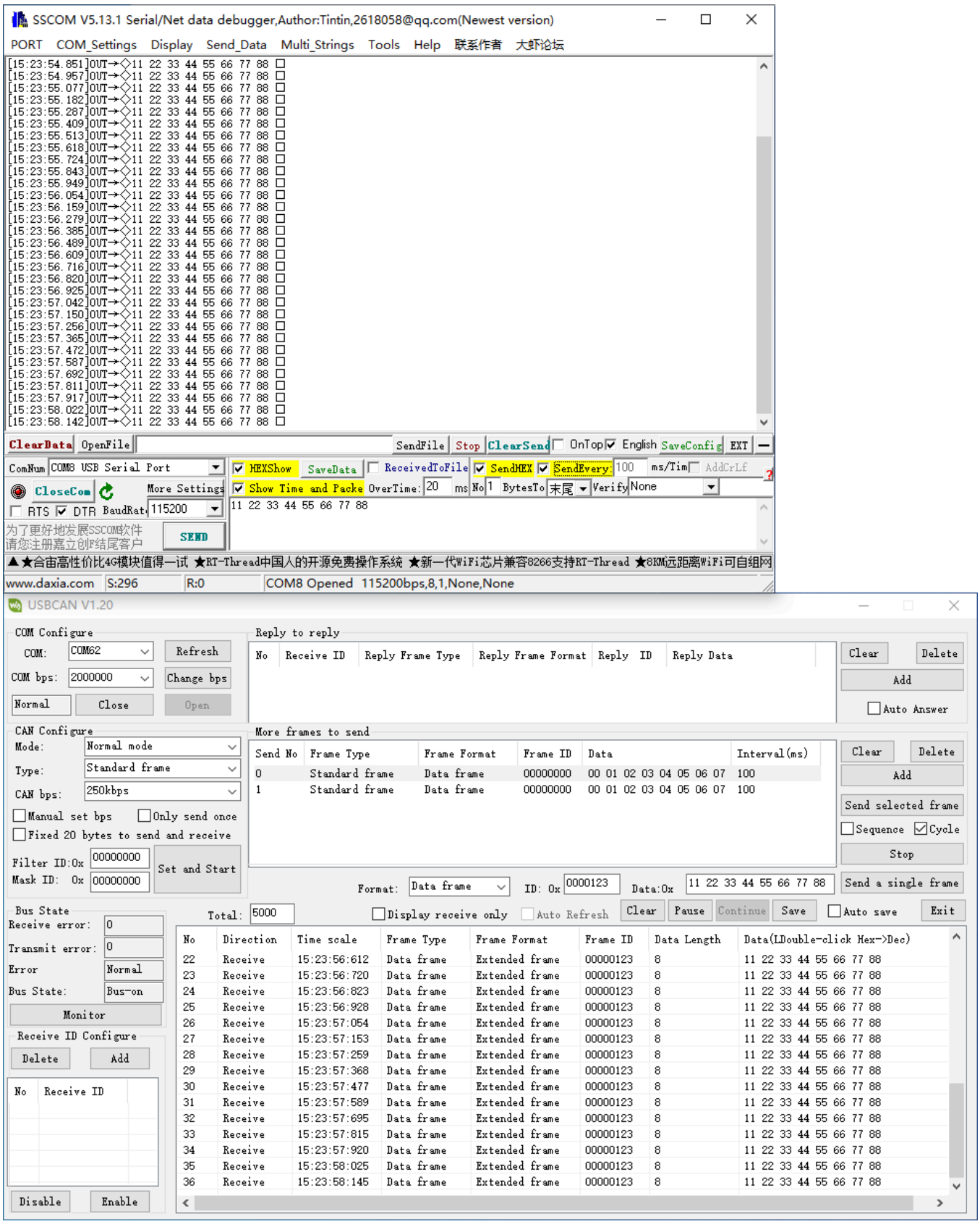
(/wiki/File:RS232_Software_Operation-4.png)

- Select data in USB-CAN-A_TOOL, click on "**Send selected frame**".



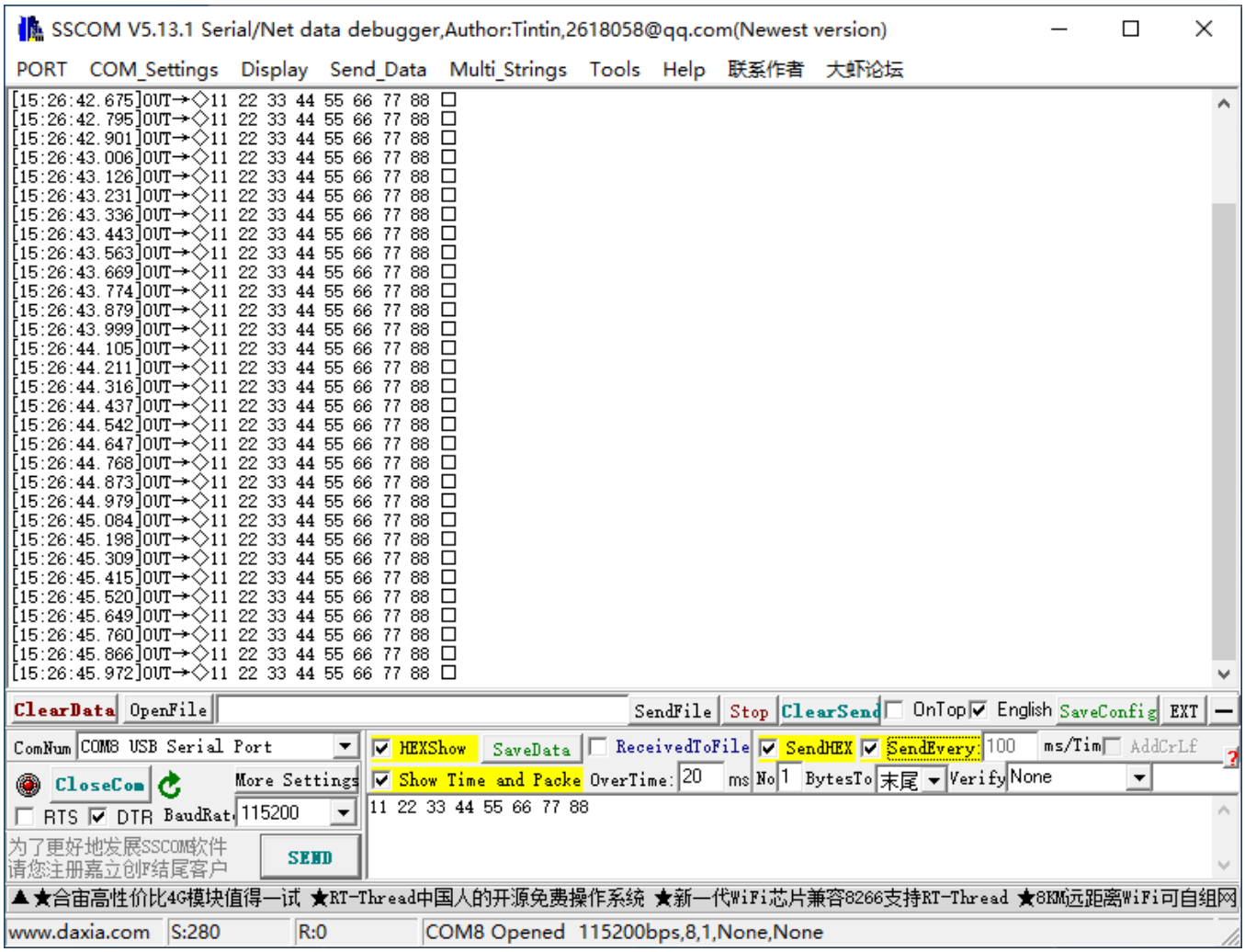
(/wiki/File:RS232_Software_Operation-5.png)

- And then you can see that it transmits and receives.



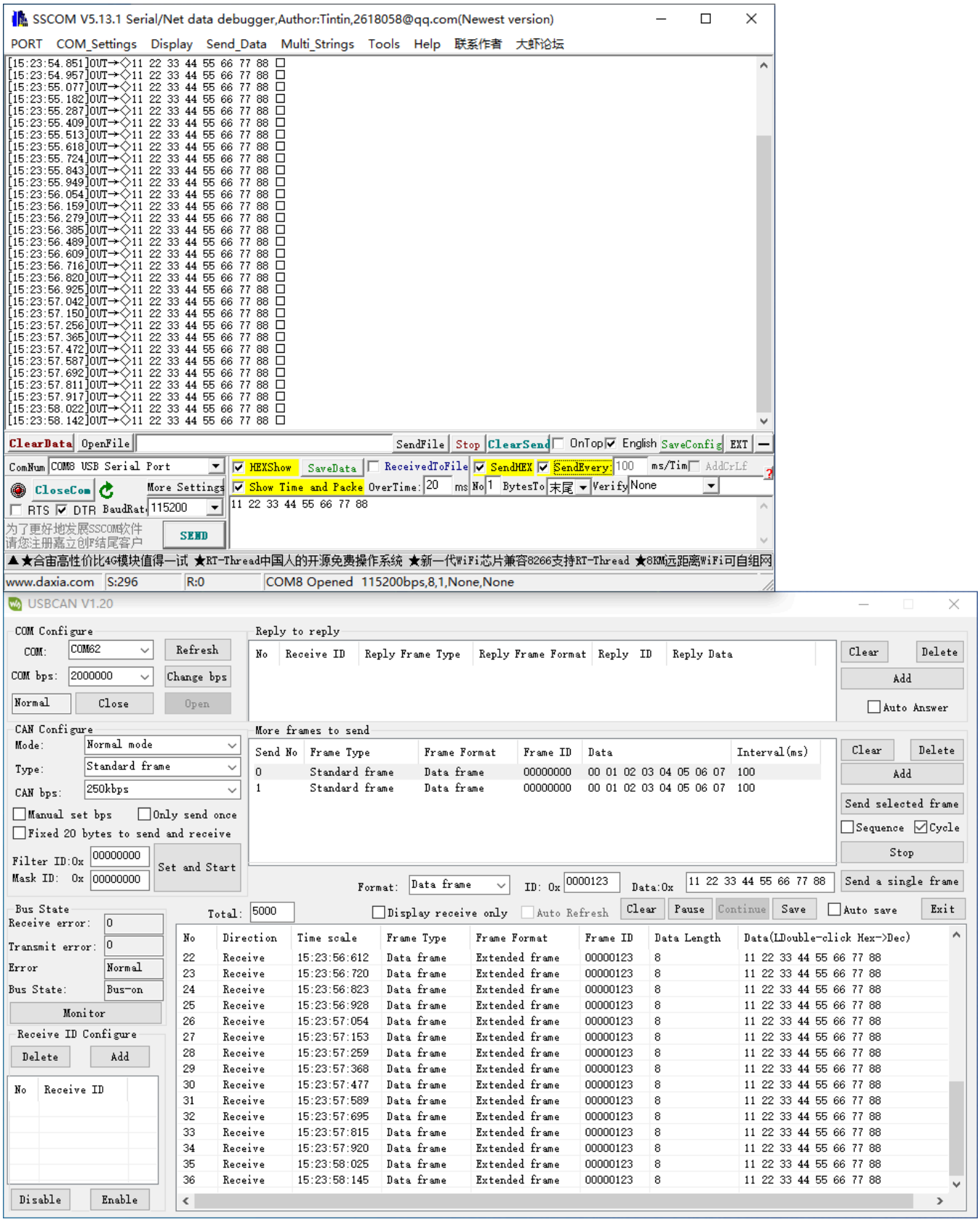
(/wiki/File:RS232_Software_Operation-6.png)

- Or you can click on "Send every...ms/Time".



(/wiki/File:RS232_Software_Operation-7.png)

- And then you can see that it transmits and receives.



(/wiki/File:RS232_Software_Operation-6.png)

RS485 TO CAN Test

Use USB-CAN-A (https://www.waveshare.com/usb-can-a.htm) and USB TO 4CH Serial

Converter (https://www.waveshare.com/usb-to-4ch-serial-converter.htm) to demonstrate.

Hardware Connection

- Connect to the CAN device.

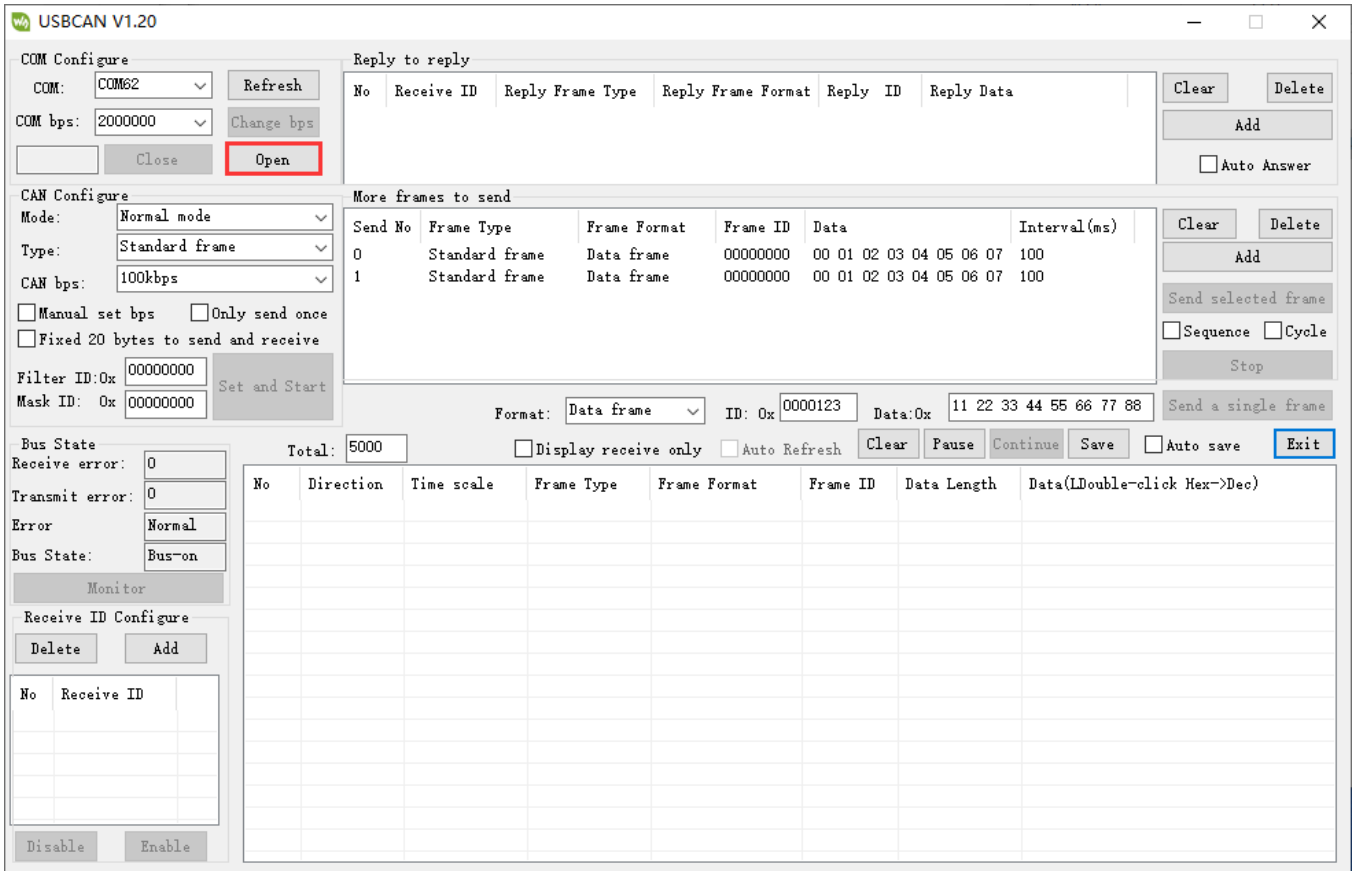
RS232/485/422 TO CAN - CAN	USB-CAN-A
CAN - CAN H	CAN - CAN H
CAN - CAN L	CAN - CAN L
CAN - CAN G	CAN - GND

- Connect to the RS485 device.

RS232/485/422 TO CAN - RS485	USB TO 4CH Serial Converter - Port C
RS485 - GND	Port C - GND
RS485 - A+	Port C - A+
RS485 - B-	Port C - B-

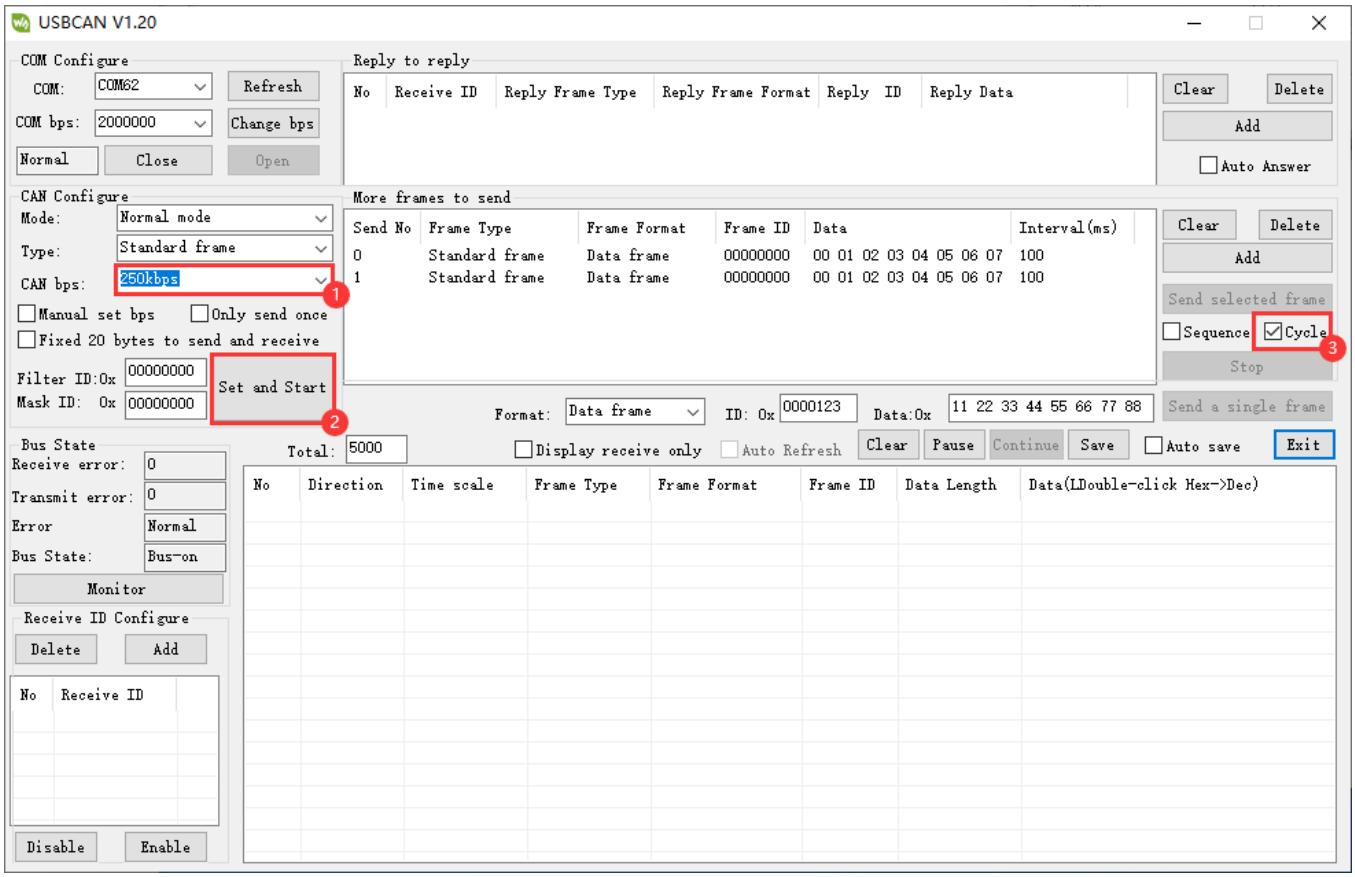
Software Operation

- Open USB-CAN-A-TOOL (https://files.waveshare.com/wiki/RS232-485-422-TO-CAN/USB-CAN-A_TOOL.zip).
- Select the corresponding COM port of the USB-CAN-A, and click on "**Open**".



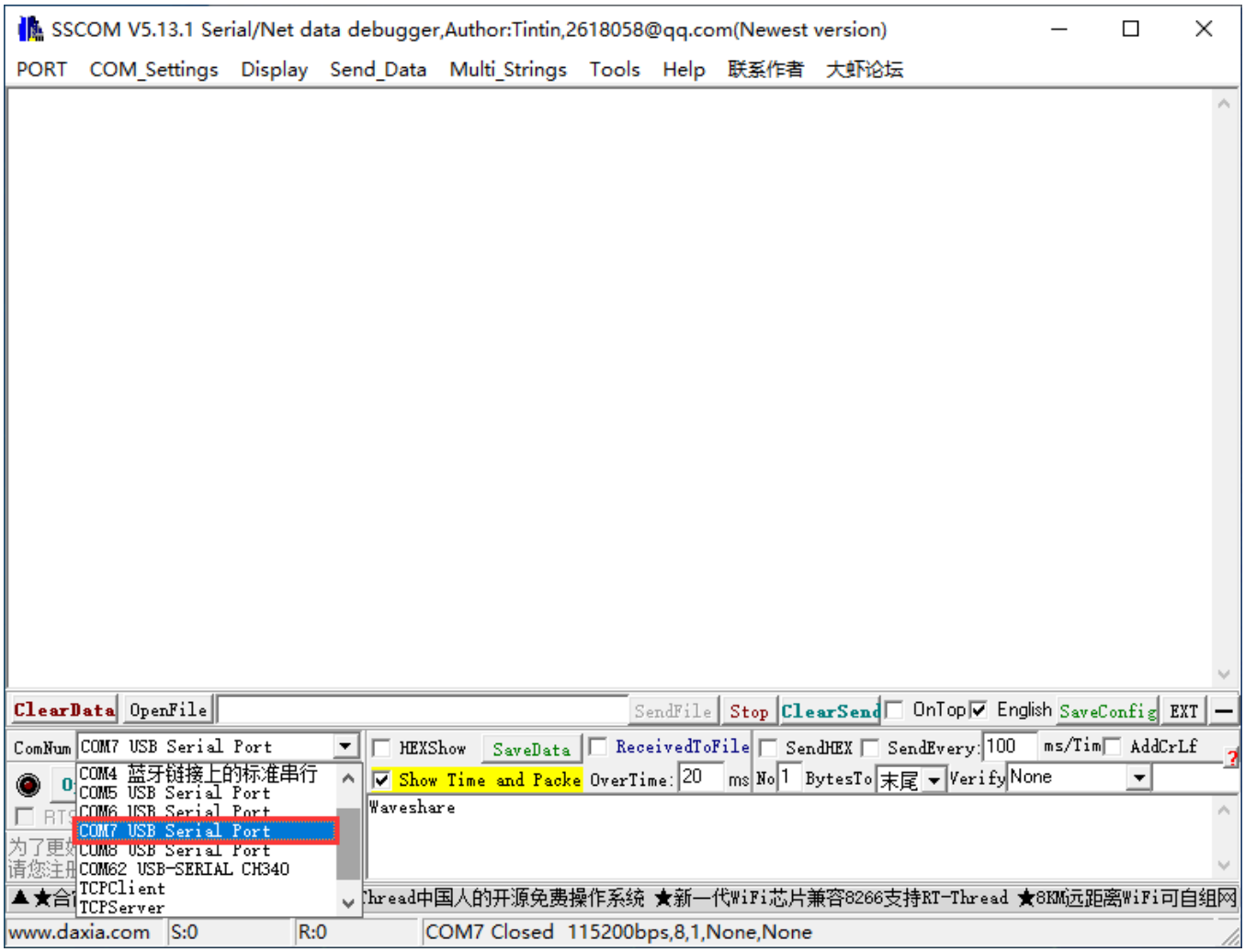
(/wiki/File:RS232_Software_Operation-1.png)

- Select the baud rate of the CAN device as 250kbps, click on "**Set and Start**" , input the data to send, and select "**Cycle**".



(/wiki/File:RS232_Software_Operation-2.png)

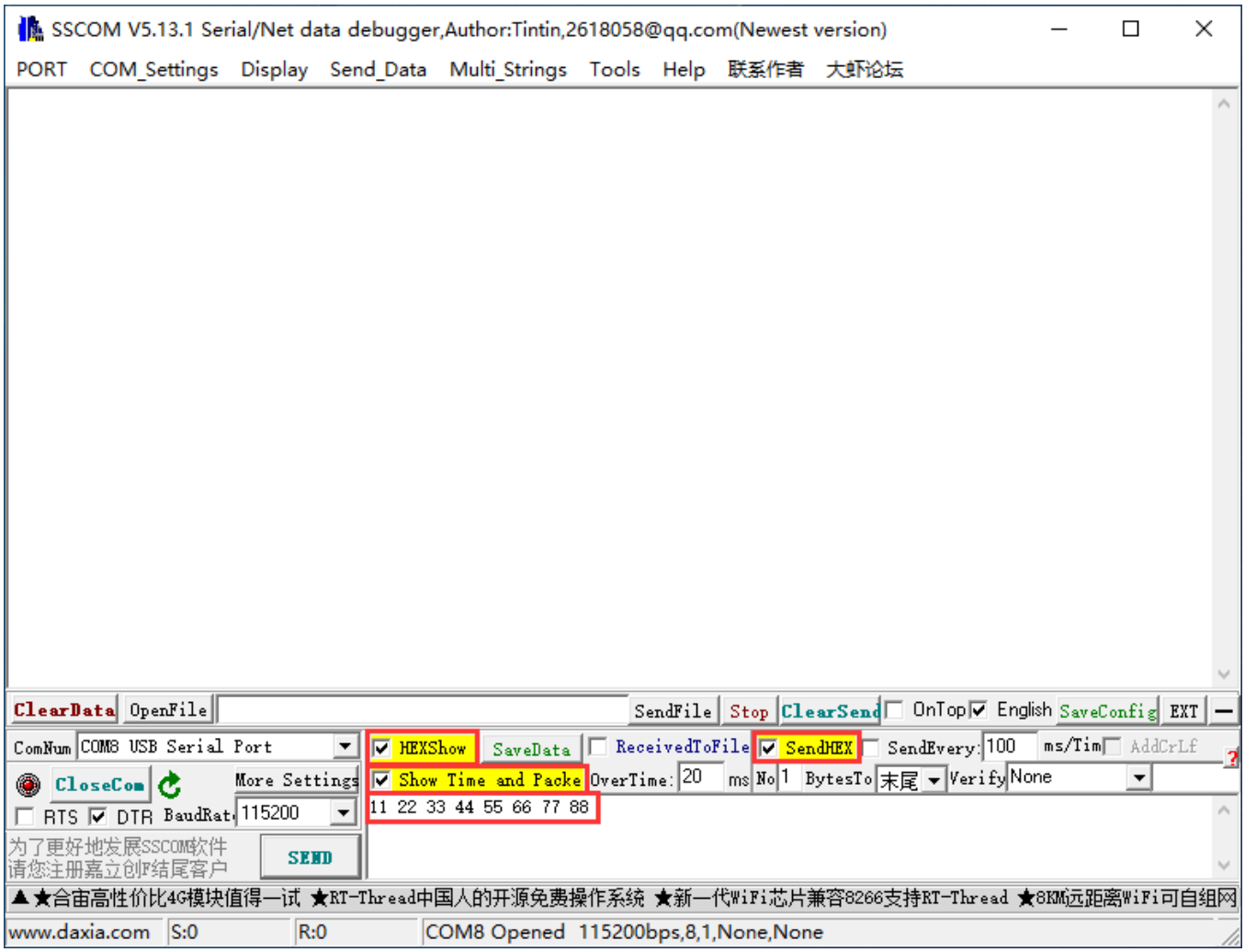
- Open an SSCOM, select the corresponding COM port of the USB TO 4CH Serial Converter's Port C, and open the serial port.



(/wiki/File:RS232_Software_Operation_03.png)

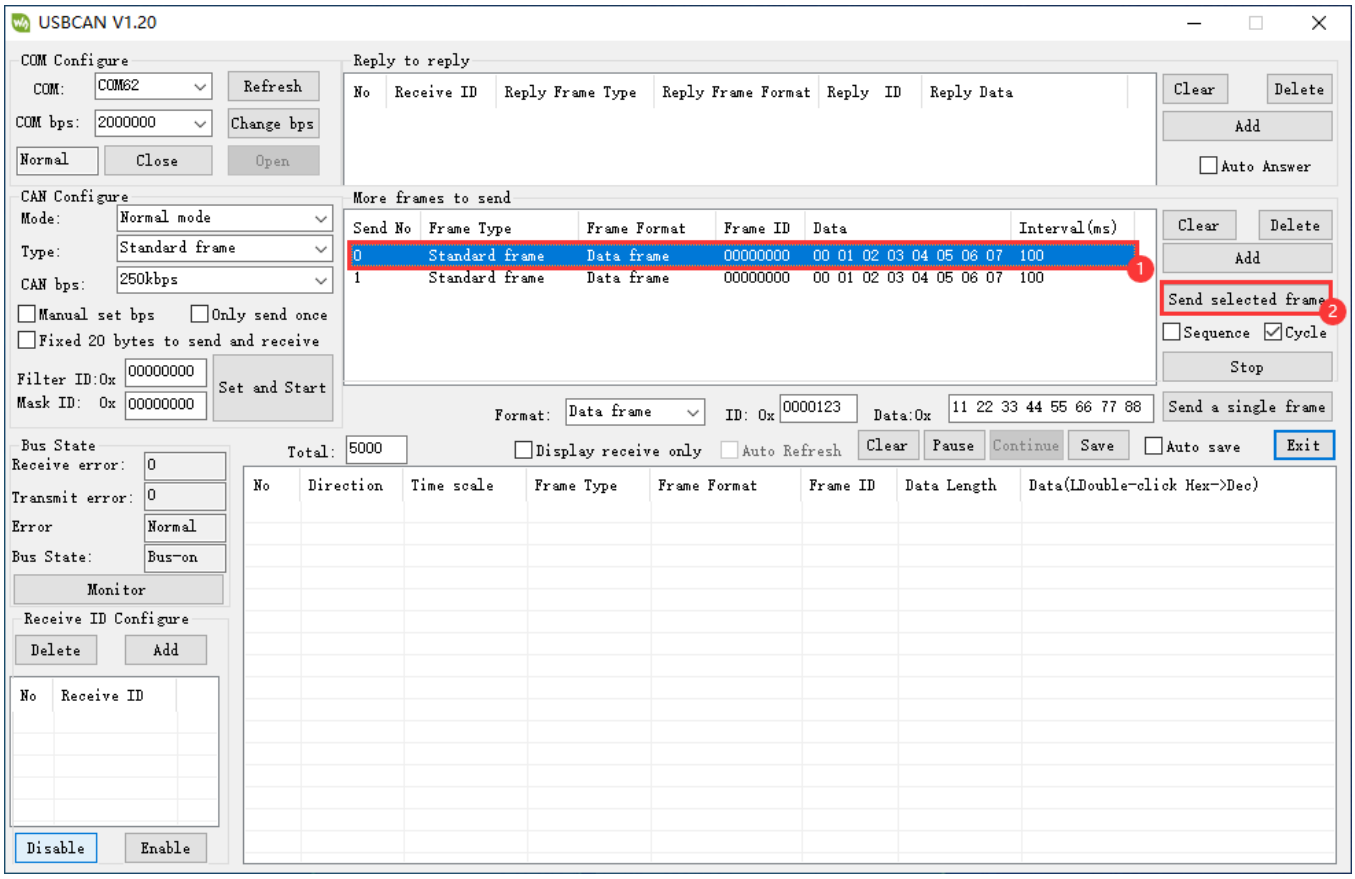
- In the SSCOM software, check "Hexshow", "sendHex" and "Show Time and Package", input the data to be sent.

```
11 22 33 44 55 66 77 88
```



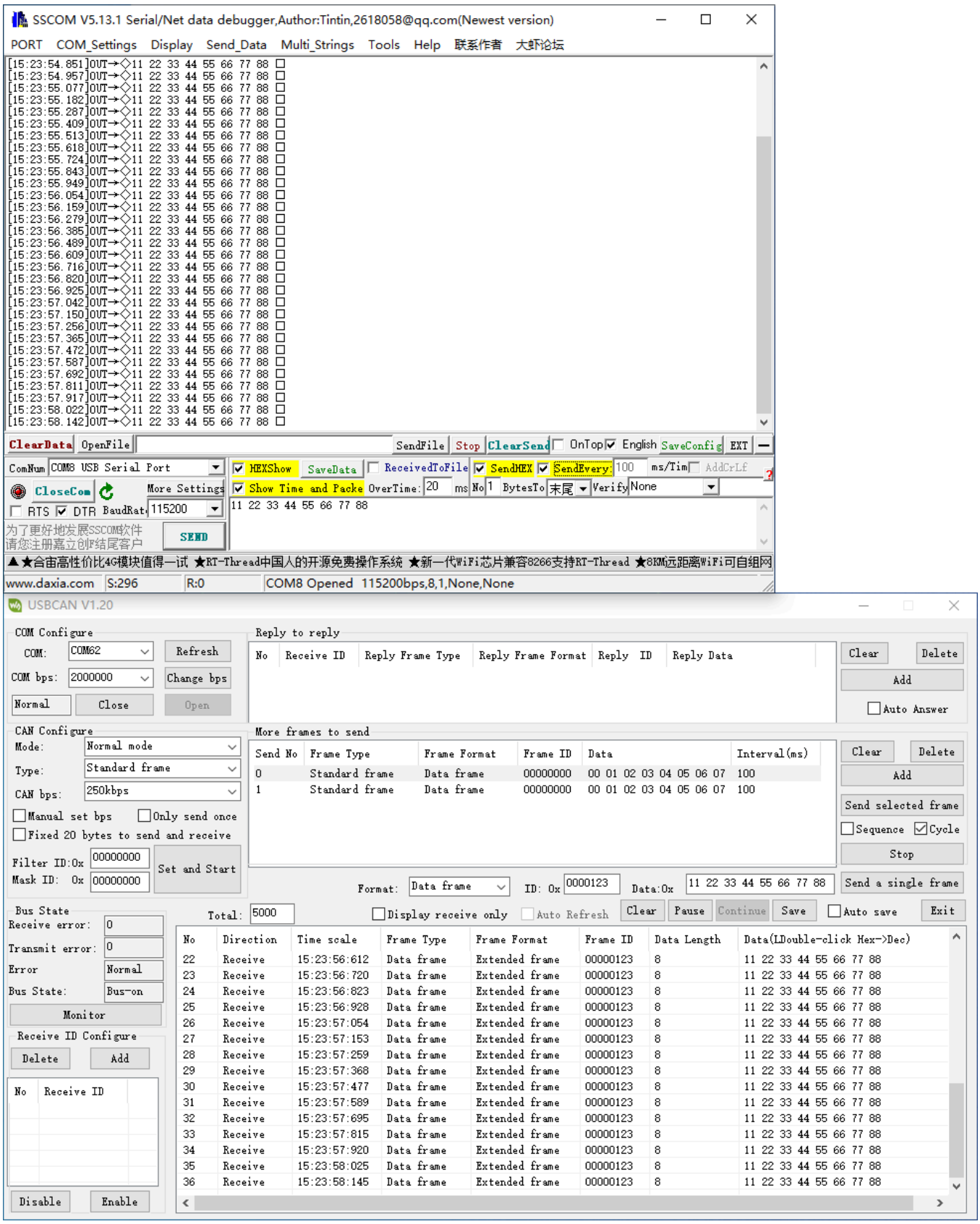
(/wiki/File:RS232_Software_Operation-4.png)

- Select data in USB-CAN-A_TOOL, click on "**Send selected frame**".



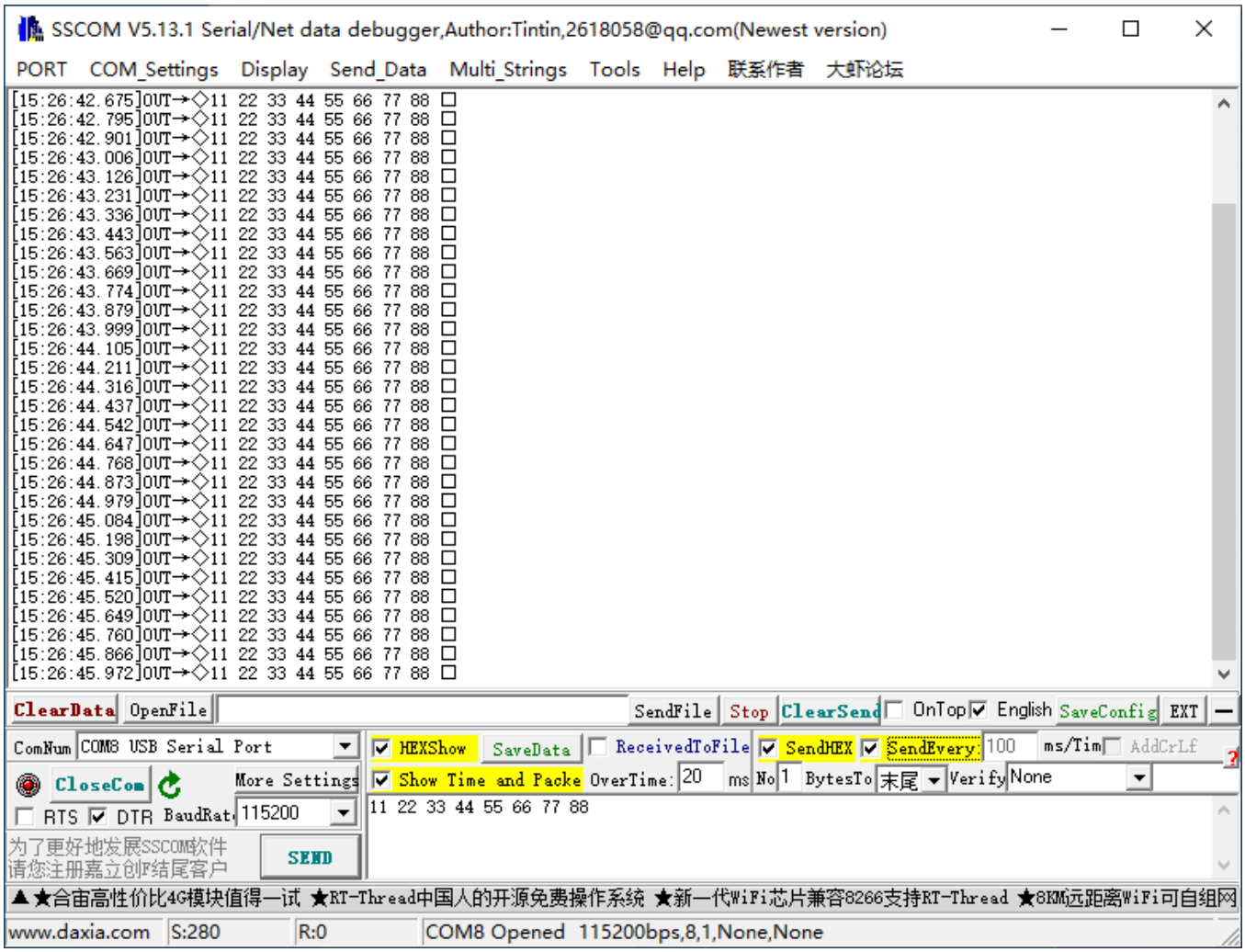
(/wiki/File:RS232_Software_Operation-5.png)

- And then you can see that it transmits and receives.



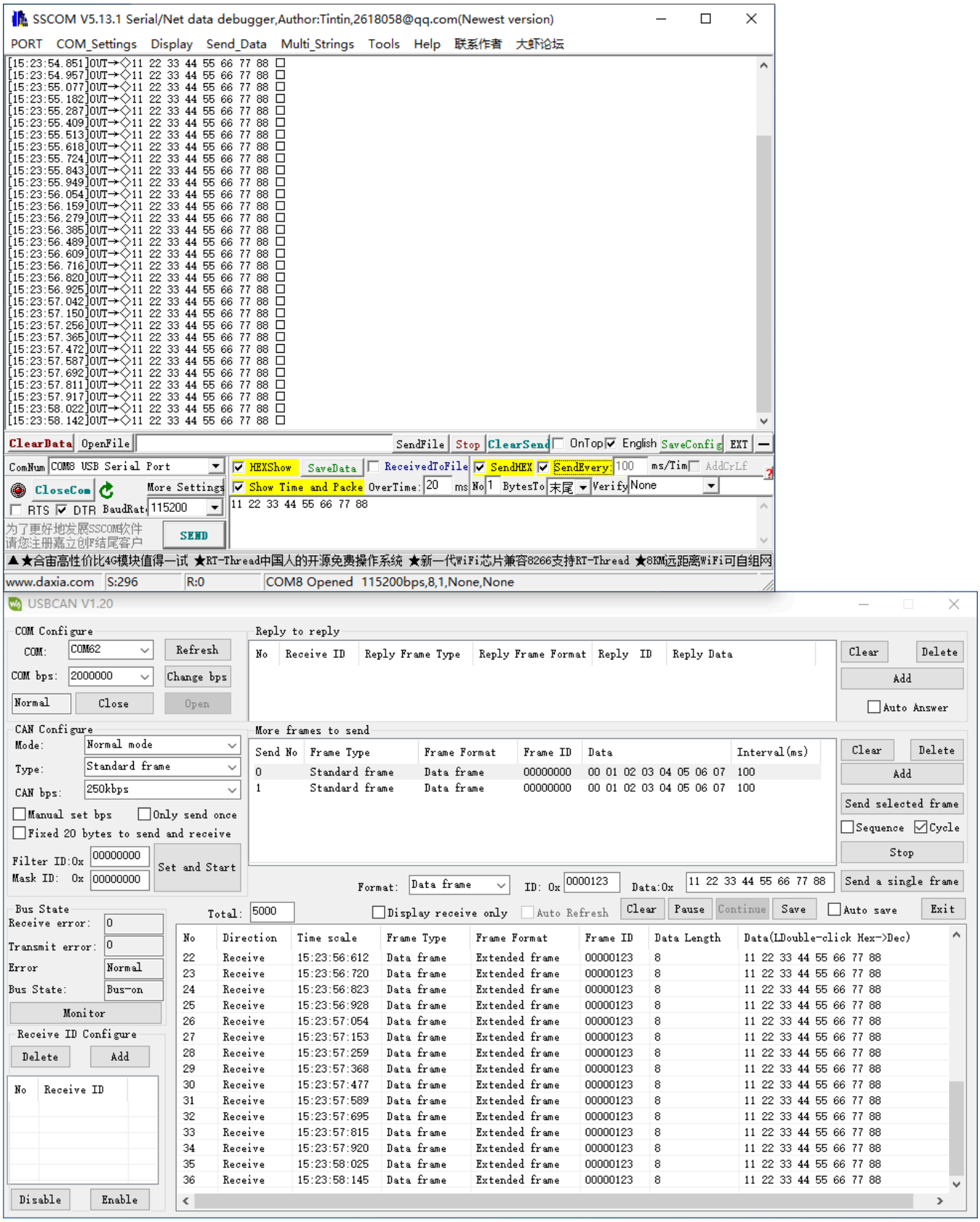
(/wiki/File:RS232_Software_Operation-6.png)

- Or you can click on "Send every...ms/Time".



(/wiki/File:RS232_Software_Operation-7.png)

- And then you can see that it transmits and receives.



(/wiki/File:RS232_Software_Operation-6.png)

RS422 TO CAN Test

Use USB-CAN-A (https://www.waveshare.com/usb-can-a.htm) and USB TO 4CH Serial Converter (https://www.waveshare.com/usb-to-4ch-serial-converter.htm) to test the

functions.

Hardware Connection

- Connect to CAN device:

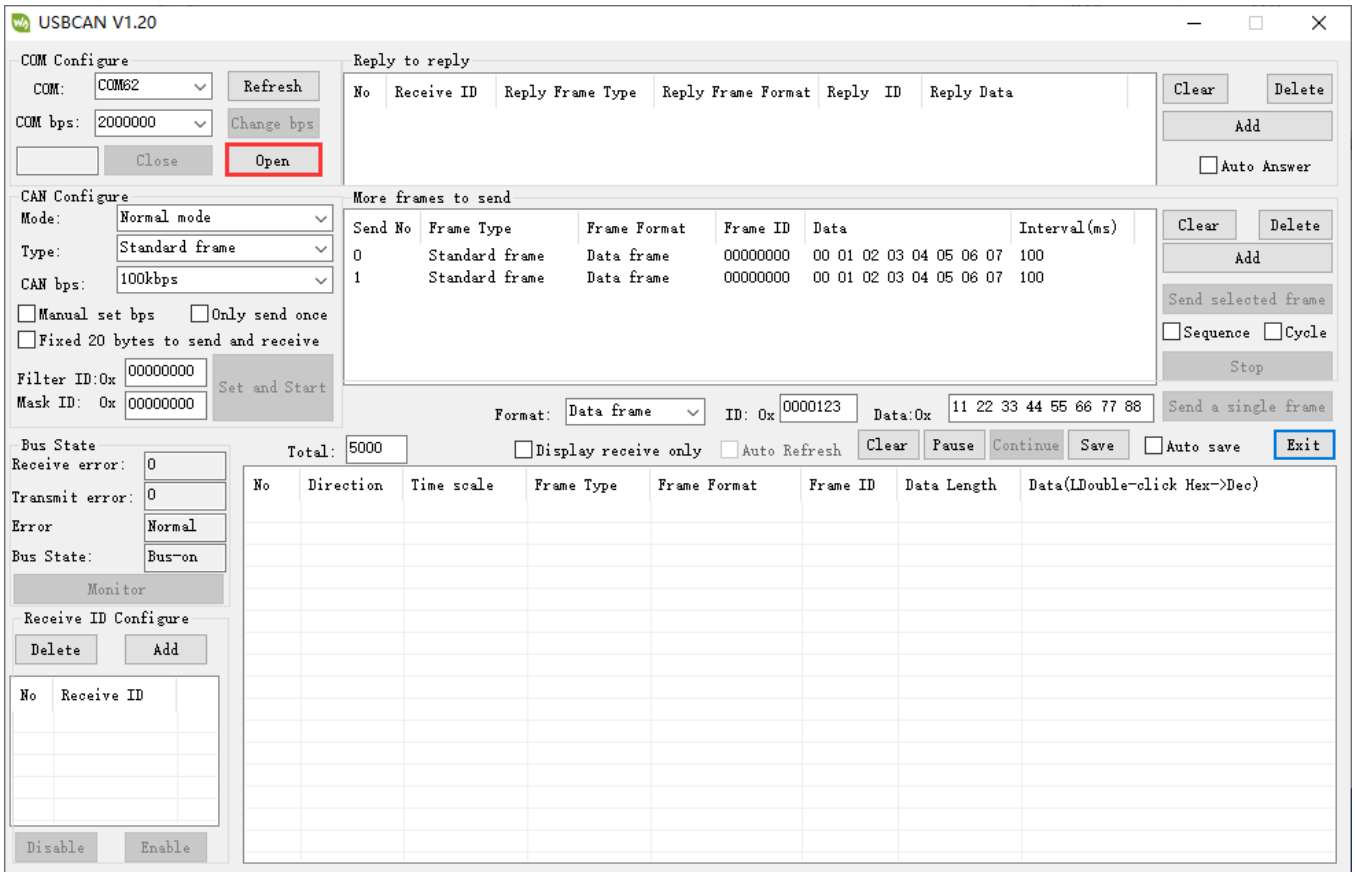
RS232/485/422 TO CAN - CAN	USB-CAN-A
CAN - CAN H	CAN - CAN H
CAN - CAN L	CAN - CAN L
CAN - CAN G	CAN - GND

- Connect to RS422 device

RS232/485/422 TO CAN - RS422	USB TO 4CH Serial Converter - Port C
RS422 - GND	Port C - GND
RS422 - TA	Port C - RA
RS422 - TB	Port C - RB
RS422 - RA	Port C - TA
RS422 - RB	Port C - TB

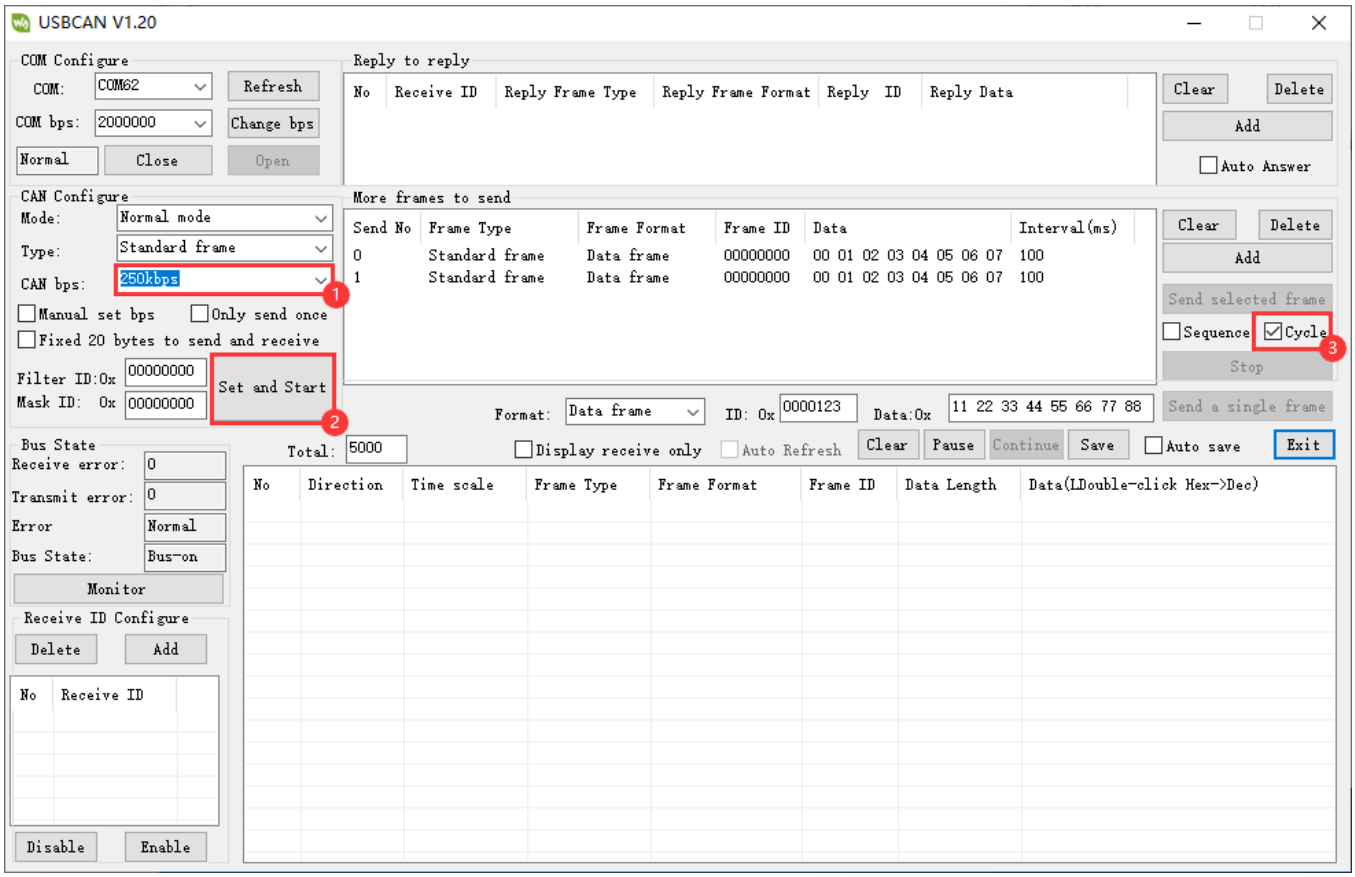
Software Operation

- Open USB-CAN-A-TOOL (https://files.waveshare.com/wiki/RS232-485-422-TO-CAN/USB-CAN-A_TOOL.zip).
- Select the corresponding COM port of the USB-CAN-A, and click on **"Open"**.



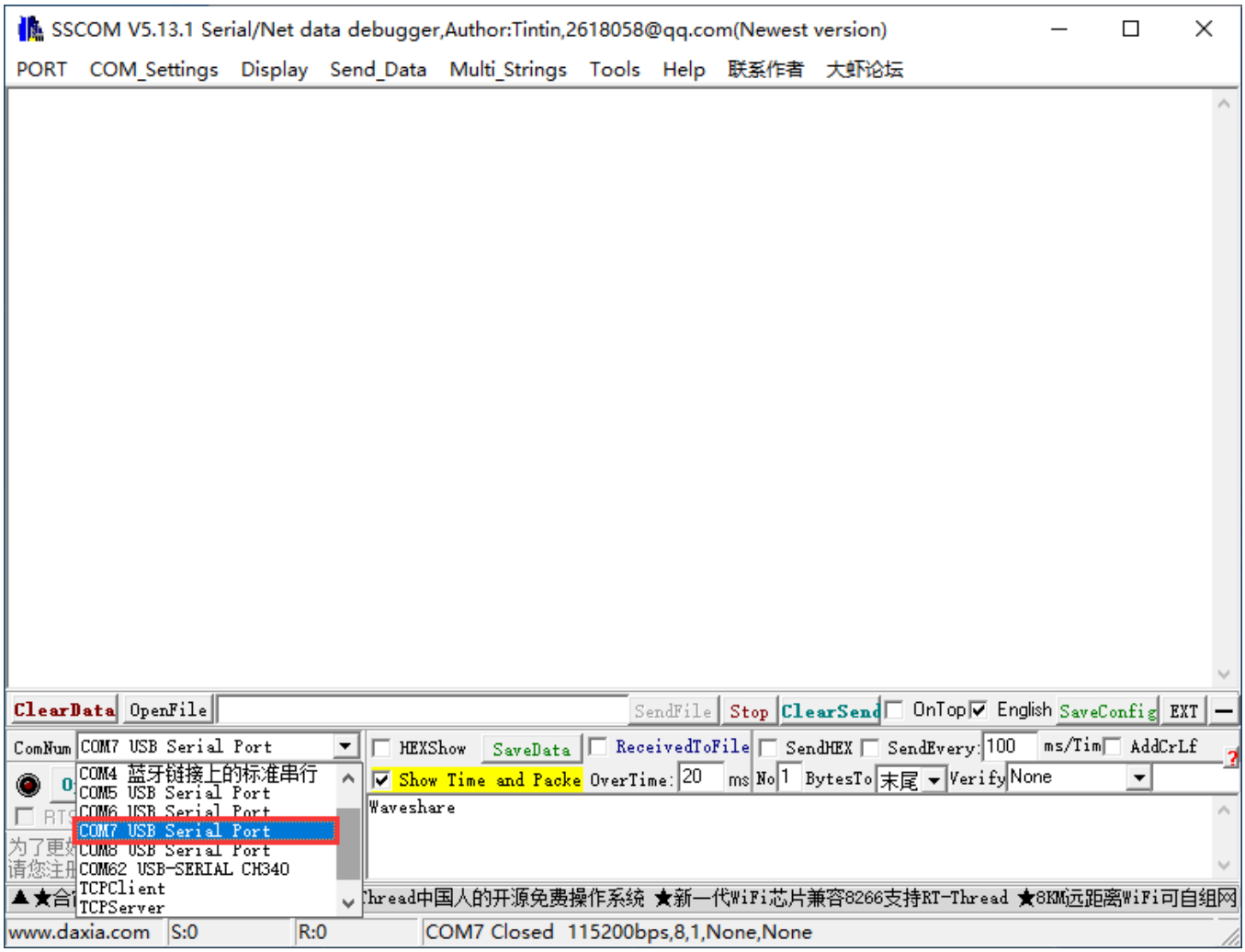
(/wiki/File:RS232_Software_Operation-1.png)

- Select the baud rate of the CAN device as 250kbps, click on **"Set and Start"** , input the data to send, and select **"Cycle"**.



(/wiki/File:RS232_Software_Operation-2.png)

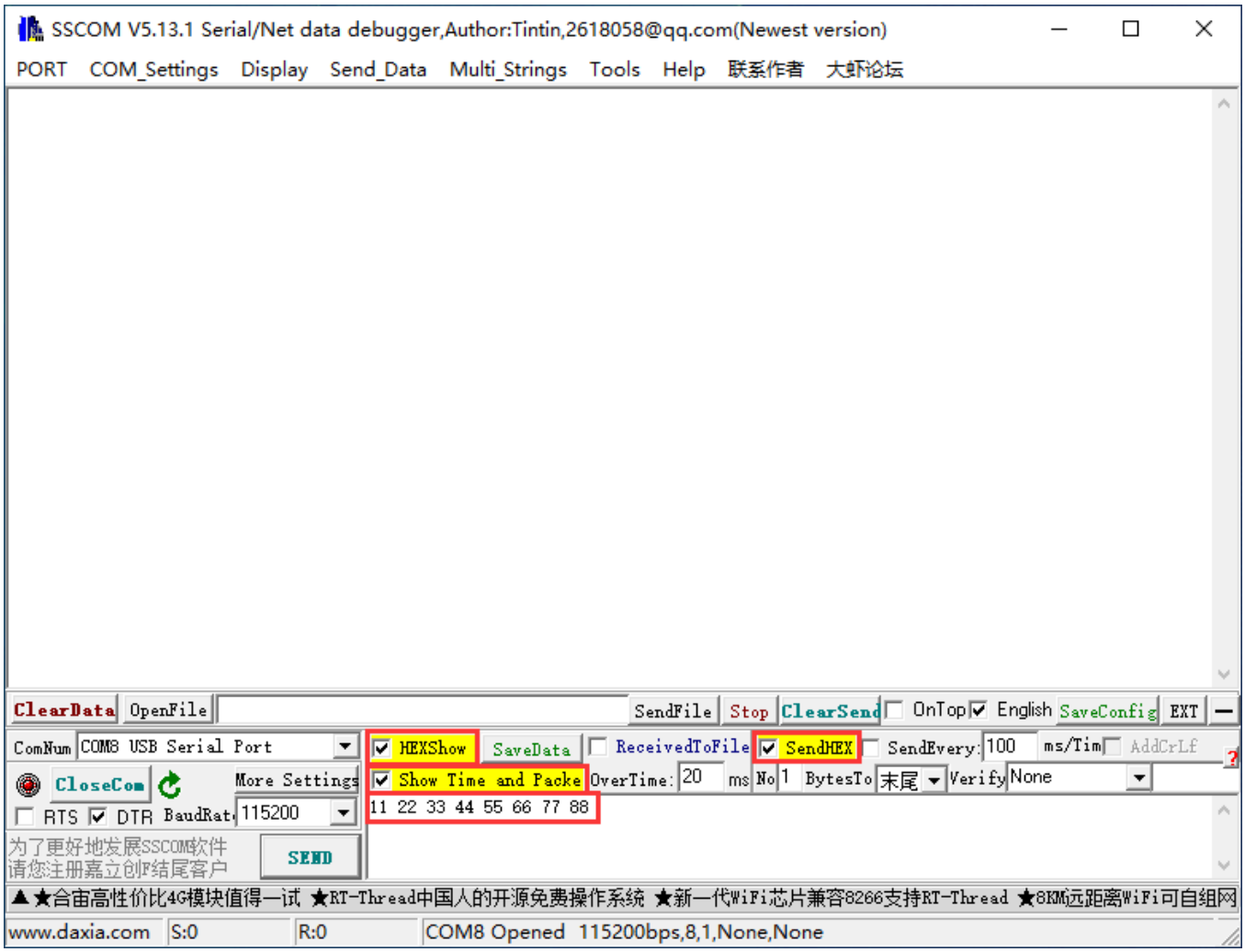
- Open an SSCOM, select the corresponding COM port of the USB TO 4CH Serial Converter's Port C, and open the serial port.



(/wiki/File:RS232_Software_Operation_03.png)

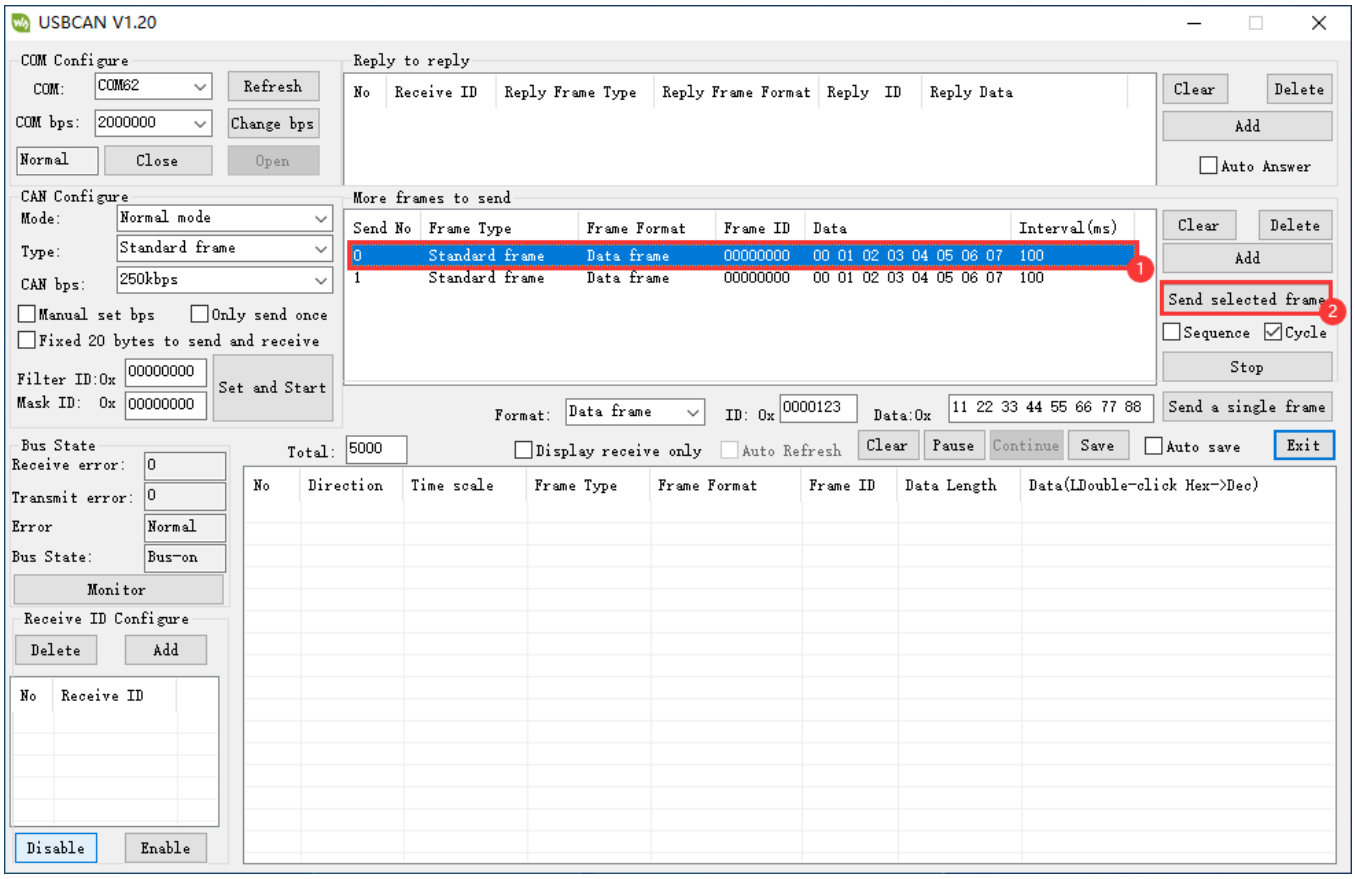
- In the SSCOM software, check "Hexshow", "sendHex" and "Show Time and Package", input the data to be sent.

```
11 22 33 44 55 66 77 88
```



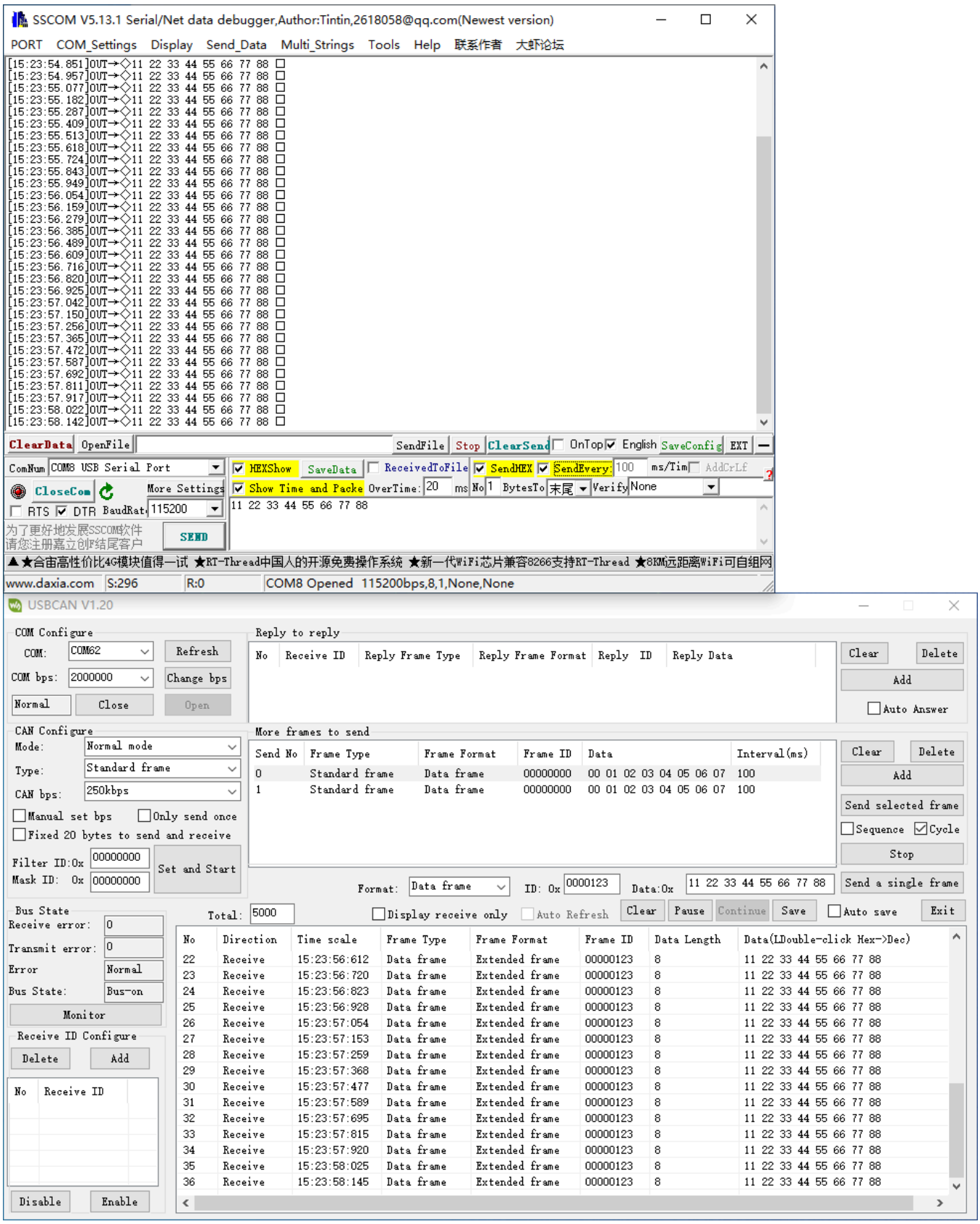
(/wiki/File:RS232_Software_Operation-4.png)

- Select data in USB-CAN-A_TOOL, click on "**Send selected frame**".



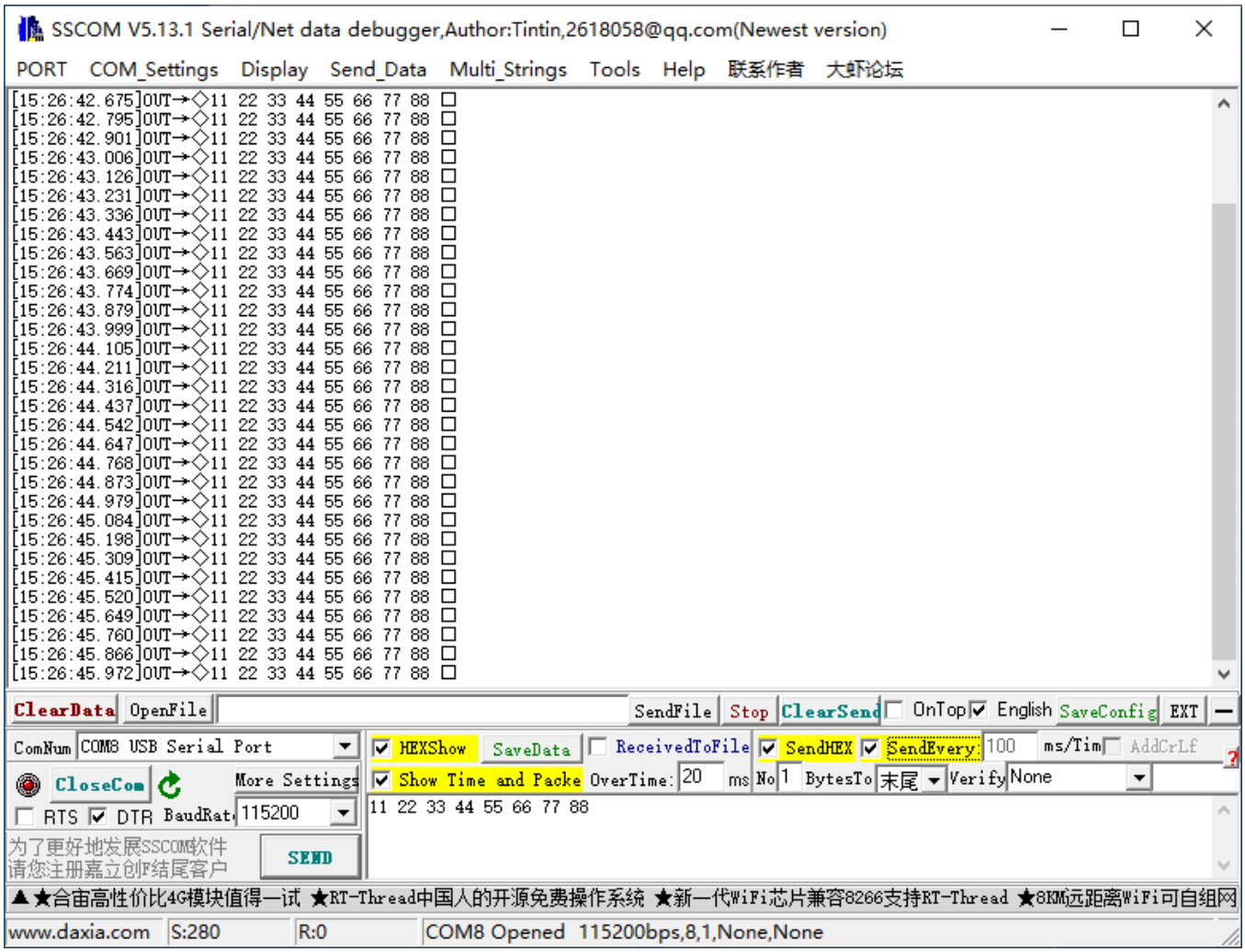
(/wiki/File:RS232_Software_Operation-5.png)

- And then you can see that it transmits and receives.



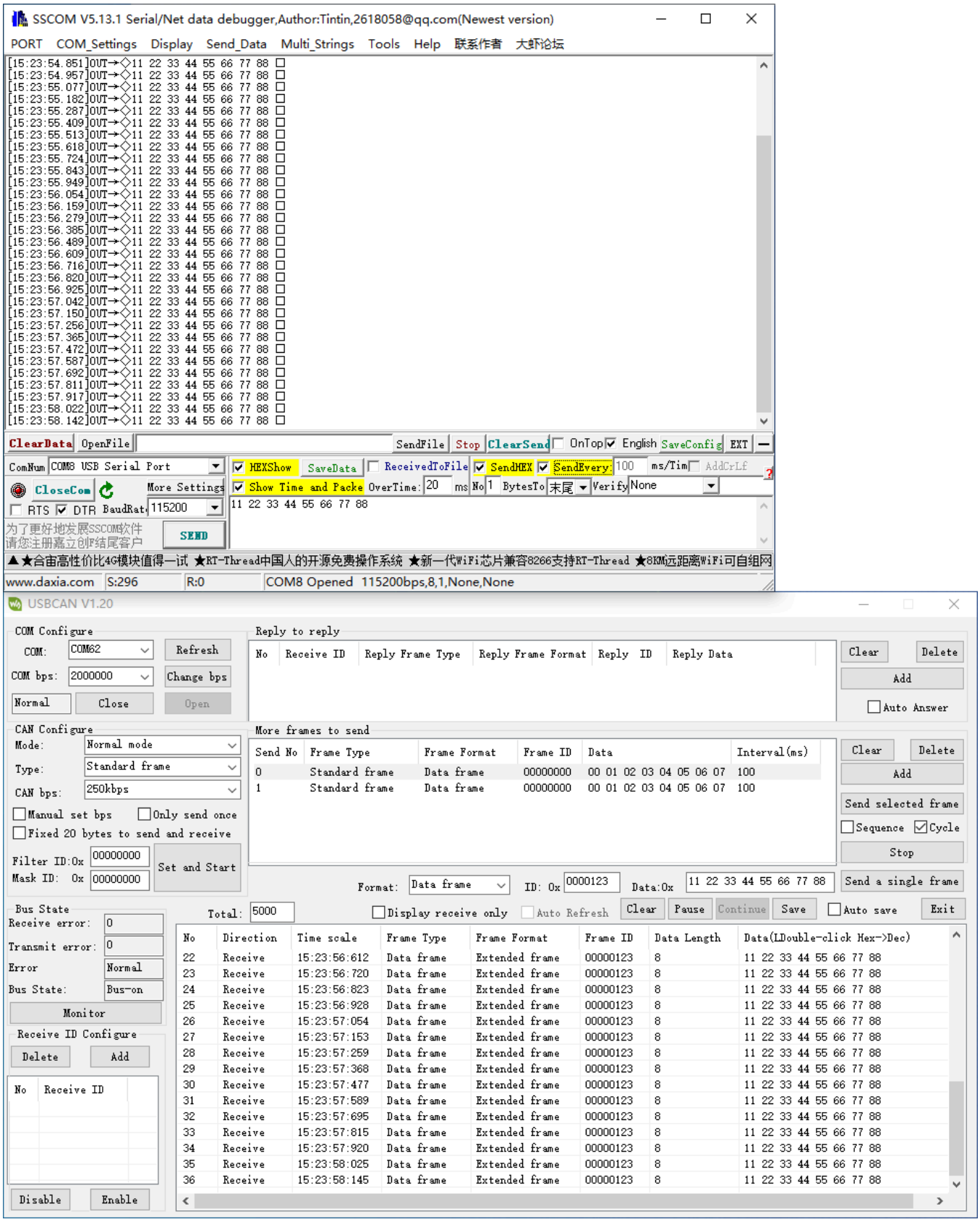
(/wiki/File:RS232_Software_Operation-6.png)

- Or you can click on "Send every...ms/Time".



(/wiki/File:RS232_Software_Operation-7.png)

- And then you can see that it transmits and receives.



(/wiki/File:RS232_Software_Operation-6.png)

RS485 CAN HAT Expansion

RS485 CAN HAT (https://www.waveshare.com/rs485-can-hat.htm) Related User's Guide (https://www.waveshare.com/wiki/RS485_CAN_HAT).

Please understand the usage of RS485 CAN HAT first.

Interface Expansion

CAN Interface Expansion

- Connect to RS485 interface, and the RS485 CAN HAT expands as 2x CAN interfaces.

RS232/485/422 TO CAN - RS485	RS485 CAN HAT
RS485 - A+	RS485 - A+
RS485 - B-	RS485 - B-

RS232/485/422 Interface Expansion

- Connect to the CAN interface, and then the RS485 CAN HAT expands as 1x RS485 interface and 1x RS232/RS485/422 interface.

RS232/485/422 TO CAN - CAN	RS485 CAN HAT
CAN - CAN H	CAN - CAN H
CAN - CAN L	CAN - CAN L

Example

Use RS232/485/422 TO CAN for the communication between RS485 and CAN of RS485 CAN HAT.

Hardware Connection

RS232/485/422 TO CAN - RS485	RS485 CAN HAT
CAN - CAN H	CAN - CAN H
CAN - CAN L	CAN - CAN L
RS485 - A+	RS485 - A+
RS485 - B-	RS485 - B-

Software Operation

Preparation

- Connect the RS485 CAN HAT to the Raspberry Pi, and modify the boot script "config.txt".

```
sudo nano /boot/config.txt
```

- Add the following content at the last line:

```
dtparam=spi=on
dtoverlay=mcp2515-can0,oscillator=12000000,interrupt=25,spimaxfrequency=2000000
```

- Also find the following configuration statement in /boot/config.txt to enable the serial port, or add it to the end of the file if it is not there.

```
enable_uart=1
```

```
  GNU nano 5.4                                /boot/config.txt
camera_auto_detect=1

# Automatically load overlays for detected DSI displays
display_auto_detect=1

# Enable DRM VC4 V3D driver
dtoverlay=vc4-kms-v3d
max_framebuffers=2

# Disable compensation for displays with overscan
disable_overscan=1

[cm4]
# Enable host mode on the 2711 built-in XHCI USB controller.
# This line should be removed if the legacy DWC2 controller is required
# (e.g. for USB device mode) or if USB support is not required.
otg_mode=1

[all]

[pi4]
# Run as fast as firmware / board allows
arm_boost=1

[all]
dtparam=spi=on
dtoverlay=mcp2515-can0,oscillator=12000000,interrupt=25,spimaxfrequency=2000000
enable_uart=1

^G Help      ^O Write Out  ^W Where Is   ^K Cut       ^T Execute   ^C Location   M-U Undo
^X Exit      ^R Read File  ^\ Replace    ^U Paste     ^J Justify   ^_ Go To Line M-E Redo
```

(/wiki/File:RS232_Preparation-3.png)

Install Library

The following operation is required.

Install can-units

```
sudo apt-get install can-utils
```

Install wiringpi environment

```
#Open the Raspberry Pi terminal and run the following commands
cd
sudo apt-get install wiringpi
#For Raspberry Pi systems after May 2019 (earlier may not be performed), an upgrade may b
e required:
wget https://project-downloads.drogon.net/wiringpi-latest.deb
sudo dpkg -i wiringpi-latest.deb
gpio -v
#Running gpio -v will bring up version 2.52, if it doesn't it means there was an installa
tion error
```

Install Python Environment

```
sudo apt-get update
sudo apt-get install python3-pip
sudo apt-get install python-serial
```

- Configure the running environment

```
sudo pip install python-can
```

- If running the above commands fails, you can download the following image:

```
sudo pip install -i https://pypi.tuna.tsinghua.edu.cn/simple (https://pypi.tuna.tsinghua.edu.cn/simple) python-can
sudo pip install -i https://mirrors.aliyun.com/pypi/simple (https://mirrors.aliyun.com/pypi/simple) python-can
sudo pip install -i https://pypi.mirrors.ustc.edu.cn/simple (https://pypi.mirrors.ustc.edu.cn/simple) python-can
```

Modify RS232/485/422 TO CAN Configuration

- Use the software to modify the CAN id of the RS232/485/422 TO CAN.
- Set the CAN ID as "123", and operate it by following the #Device Configuration

C Example

- **If you have already downloaded, please skip the file deletion step and the program download step (the two steps below).**
- Delete the previously downloaded package.

Please note: If you want to keep the original RS485_CAN_HAT_Code file, please copy it to another path.

```
cd
rm -rf RS485_CAN_HAT_Code.zip
rm -rf RS232-485-422_Code.zip
rm -rf RS485_CAN_HAT_Code
sudo ifconfig can0 down
```

- Download the example demo:

```
cd
sudo apt-get install unzip
wget https://files.waveshare.com/wiki/RS232-485-422-TO-CAN/RS232-485-422_Code.zip
unzip RS232-485-422_Code.zip
sudo chmod 777 -R RS485_CAN_HAT_Code/
```

- CAN sends and the RS485 receives:

```
cd
cd RS485_CAN_HAT_Code/CAN_TO_RS485/wiringPi/
make clean
make
sudo ./can_send
```

```
pi@raspberrypi:~ $ cd
cd RS485_CAN_HAT_Code/CAN_TO_RS485/wiringPi/
make clean
make
sudo ./can_send
rm -f can_send .*.sw?
gcc -Wall -g -O0 -o can_send can_send.c -lwiringPi
RTNETLINK answers: Device or resource busy
this is a can send demo
set wiringPi lib success  !!!
======================== NEW Data 0 ========================
Count Down :5 4 3 2 1
can_id  = 0x123
can_dlc = 8
CAN Send data[0] = 0x01
CAN Send data[1] = 0x02
CAN Send data[2] = 0x03
CAN Send data[3] = 0x04
CAN Send data[4] = 0x05
CAN Send data[5] = 0x06
CAN Send data[6] = 0x07
CAN Send data[7] = 0x08

RS485 Read Data : 0x01
RS485 Read Data : 0x02
RS485 Read Data : 0x03
RS485 Read Data : 0x04
RS485 Read Data : 0x05
RS485 Read Data : 0x06
RS485 Read Data : 0x07
RS485 Read Data : 0x08
======================== NEW Data 1 ========================
Count Down :5 4 3 2 1
```

(/wiki/File:RS232_CAN_01.png)

- RS485 sends and CAN receives:

```
cd
cd RS485_CAN_HAT_Code/RS485_TO_CAN/wiringPi/
make clean
make
sudo ./485_send
```

```
======================== NEW Data 0 ========================
Count Down :5 4 3 2 1
RS485 Send Data:
RS485 Send Data[0]:11
RS485 Send Data[1]:22
RS485 Send Data[2]:33
RS485 Send Data[3]:44
RS485 Send Data[4]:55
RS485 Send Data[5]:66
RS485 Send Data[6]:77
RS485 Send Data[7]:88

CAN Read Data:
can_id = 0x123
can_dlc = 8
CAN Read data[0] = 11
CAN Read data[1] = 22
CAN Read data[2] = 33
CAN Read data[3] = 44
CAN Read data[4] = 55
CAN Read data[5] = 66
CAN Read data[6] = 77
CAN Read data[7] = 88
======================== NEW Data 1 ========================
Count Down :5 4
```

(/wiki/File:RS232_CAN_02.png)

Python Example

- **If you have already downloaded, please skip the file deletion step and the program download step (the two steps below).**
- Delete the previously downloaded package.

Please note: If you want to keep the original RS485_CAN_HAT_Code file, please copy it to another path.

```
cd
rm -rf RS485_CAN_HAT_Code.zip
rm -rf RS232-485-422_Code.zip
rm -rf RS485_CAN_HAT_Code
sudo ifconfig can0 down
```

- Download the demo:

```
cd
sudo apt-get install unzip
wget https://files.waveshare.com/wiki/RS232-485-422-TO-CAN/RS232-485-422_Code.zip
unzip RS232-485-422_Code.zip
sudo chmod 777 -R RS485_CAN_HAT_Code/
```

- Reopen the can0, CAN sends and RS485 receives:

```
cd
cd RS485_CAN_HAT_Code/CAN_TO_RS485/python/
sudo python can_send.py
```

```
pi@raspberrypi:~/RS485_CAN_HAT_Code/RS485_TO_CAN/wiringPi $ cd
cd RS485_CAN_HAT_Code/CAN_TO_RS485/python/
sudo python can_send.py
RTNETLINK answers: Device or resource busy
/home/pi/RS485_CAN_HAT_Code/CAN_TO_RS485/python/can_send.py:18: RuntimeWarning: This channel is alre
ady in use, continuing anyway.  Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(EN_485,GPIO.OUT)
======================== NEW Data 0 ========================
Count Down :5 4 3 2 1
CAN Send Data: 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07

RS485 Read Data: 0x00
RS485 Read Data: 0x01
RS485 Read Data: 0x02
RS485 Read Data: 0x03
RS485 Read Data: 0x04
RS485 Read Data: 0x05
RS485 Read Data: 0x06
RS485 Read Data: 0x07
======================== NEW Data 1 ========================
Count Down :5 4 3 2 1
```

(/wiki/File:RS232_CAN_03.png)

- Reopen the can0, RS485 sends and CAN receives:

```
cd
cd RS485_CAN_HAT_Code/RS485_TO_CAN/python/
sudo python 485_send.py
```

```
pi@raspberrypi:~/RS485_CAN_HAT_Code/CAN_TO_RS485/python $ cd
cd RS485_CAN_HAT_Code/RS485_TO_CAN/python/
sudo python 485_send.py
RTNETLINK answers: Device or resource busy
/home/pi/RS485_CAN_HAT_Code/RS485_TO_CAN/python/485_send.py:18: RuntimeWarning: This channel is alre
ady in use, continuing anyway.  Use GPIO.setwarnings(False) to disable warnings.
  GPIO.setup(EN_485,GPIO.OUT)
======================== NEW Data 0 ========================
Count Down :5 4 3 2 1
RS485 Send Data: 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08
CAN Read Byte: 0x01
CAN Read Byte: 0x02
CAN Read Byte: 0x03
CAN Read Byte: 0x04
CAN Read Byte: 0x05
CAN Read Byte: 0x06
CAN Read Byte: 0x07
CAN Read Byte: 0x08
======================== NEW Data 1 ========================
Count Down :5
```

(/wiki/File:RS232_CAN_04.png)

Resource

Software

- USB-CAN-A_TOOL_1.2_EN (https://files.waveshare.com/upload/8/8a/USB-CAN-A_TOOL_EN.zip)
- WS-CAN-TOOL (https://files.waveshare.com/wiki/WS-TTL-CAN/tool/WS-CAN-TOOL%20(17).zip)

- SSCOM (https://files.waveshare.com/wiki/RS232-485-422-TO-CAN/Cktszsss32.zip)

External Test Demo

- RS485 CAN HAT Demo (https://files.waveshare.com/wiki/RS232-485-422-TO-CAN/RS232-485-422_Code.zip)

Datasheet

- SP3232EEN (https://files.waveshare.com/wiki/RS232-485-422-TO-CAN/SP3232EEN.pdf)
- SP3485EN (https://files.waveshare.com/wiki/RS232-485-422-TO-CAN/SP3485EN.pdf)
- RS232/485/422 TO CAN User Manual (https://files.waveshare.com/wiki/RS232-485-422-TO-CAN/RS232-485-422-TO-CAN-User-Manual.pdf)

FAQ

Question: How to determine whether the CAN to Modbus RTU function is suitable for my use case?

Answer:

If your requirement is to "transparently tunnel complete CAN frames into Modbus RTU data frames" or to "convert CAN bus data into Modbus registers and have a Modbus master poll these registers to retrieve CAN data", then this function will not meet your needs.

The CAN to Modbus RTU function of this module only supports data forwarding in a specific format. It does not provide full CAN frame transparent tunneling, nor does it offer storage, buffering, or mapping of CAN messages to Modbus registers. Therefore, this module cannot act as a standard Modbus slave for a Modbus master to poll and retrieve CAN bus data via register reads.

CAN bus data will not be automatically converted into Holding Registers, Input Registers, or other register types. For complete CAN frame parsing, data buffering, register mapping, or master polling, you need to implement these functions on the host PC, PLC, or in your own application program.

Functional limitations:
- Only the lowest 8 bits of the CAN ID are retained. For example, CAN ID `0x123` becomes `0x23` after forwarding.
- The first byte of the CAN data field is not forwarded as Modbus data, so the first byte of the CAN data field cannot be used as valid data.
- The forwarded data does not include the complete CAN ID or the complete CAN data field.

Example:

CAN frame:

123#A1 B1 B2 B3 B4 B5 B6 B7 B8

Forwarded Modbus data:

23 B1 B2 B3 B4 B5 B6 B7

Question: The configurations are all correct, but the communication is not working?

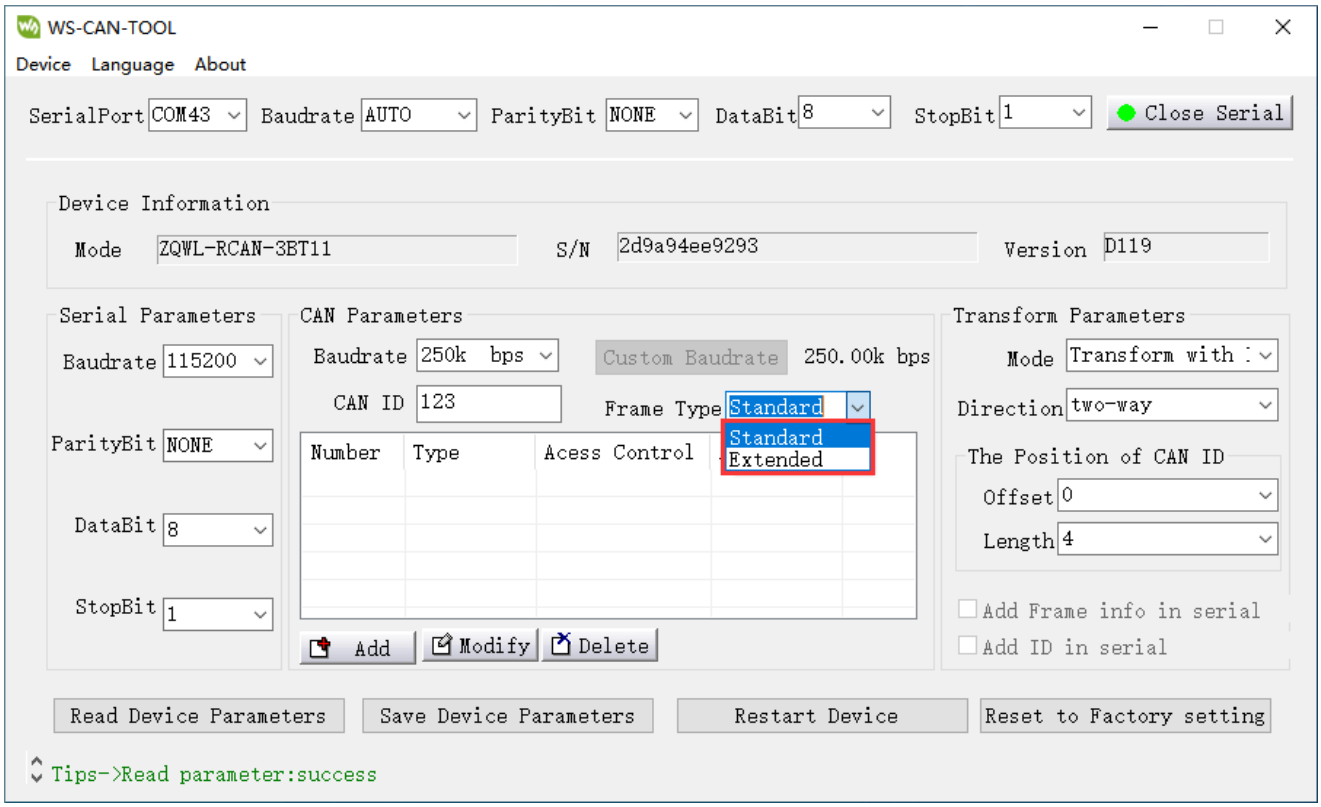
Answer:

- Please press and hold the reset button for more than 5 seconds until the indicator blinks continuously (factory restored), and then reconfigure the communication parameters as required.

Question: Why does the CAN ID only receive two bytes?

Answer:

- Please note that the received data frame type. If the received data type is an extended frame, the frame ID is **0xFFFFFFFF & CAN_ID**. If the received data type is a standard frame, the frame ID is **0x000007FF & CAN_ID**, which is not related to the frame type configured by the software (in transparent conversion mode).
- Use the one configured by the software as the frame type when sending data, as follows:



(/wiki/File:RS232_TO_CAN_FAQ02.png)

Question: Does the module has onboard 120ohm resistor?

Answer:

Yes, if you want to modify the 120R, you can move the jumper cap on the RS485/422 terminal to the 120R.

Question: What can I do if the remote communication of the RS485/422 is abnormal?

Answer:

You can move the jumper cap of the RS485/422 to the 120R.

Question: How to deal with RS485/422 short-distance communication anomalies, where extra 0s or garbled data are received?

Answer:

Do not enable the two 120Ω termination matching resistors of RS485/422, i.e., the jumper cap should be set to NC.

Question: What is the power consumption of the device?

Answer:

- Under 12V power supply, the device power consumption will fluctuate between 0.2W and 0.4W. Actual power consumption may vary depending on the specific operating conditions of the device; please refer to the actual usage for accurate values.

Question: How to use the filter settings?

Answer:

- Filter Acceptance Code

This is the reference value used for matching when a CAN frame is received. It serves as the baseline for comparing the incoming message ID.

- Filter Acceptance Mask

This defines the matching rule on a bit-by-bit basis. When a mask bit is 1, the corresponding bit in the Acceptance Code must match the message ID exactly. When a mask bit is 0, the corresponding bit is ignored (no match required).

- Example 1 (Acceptance Code 0x122 + Mask 0xFFF – exact ID match)

Set the Acceptance Code to 0x122 (binary: 0001 0010 0010) and the Acceptance Mask to 0xFFF (binary: 1111 1111 1111). All mask bits are 1, requiring every bit of the Acceptance Code to match the message ID exactly. As a result, only CAN messages with ID 0x122 are accepted; all other IDs are filtered out.

- Example 2 (Acceptance Code 0x122 + Mask 0xFFE – relaxed match)

Set the Acceptance Code to 0x122 (binary: 0001 0010 0010) and the Acceptance Mask to 0xFFE (binary: 1111 1111 1110). Only the lowest bit of the mask is 0, while all other bits are 1. This means that all bits except the lowest must match the Acceptance Code exactly; the lowest bit can be either 0 or 1. Thus, two CAN IDs are accepted:
0x122 (binary: 0001 0010 0010, lowest bit 0)
0x123 (binary: 0001 0010 0011, lowest bit 1)
All other IDs are filtered out.

Note: The above logic applies to 11-bit standard CAN IDs. All values are given in hexadecimal (with 0x prefix); binary representations are provided for clarity to illustrate the bit-matching rules.

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=RS232/485/422_TO_CAN&oldid=110899 (https://www.waveshare.com/w/index.php?title=RS232/485/422_TO_CAN&oldid=110899)"