ExUC USB CAN

Linux SocketCAN Driver

Installation Guide

Table of Contents

1. Hardware Installation

1.1. ExUC

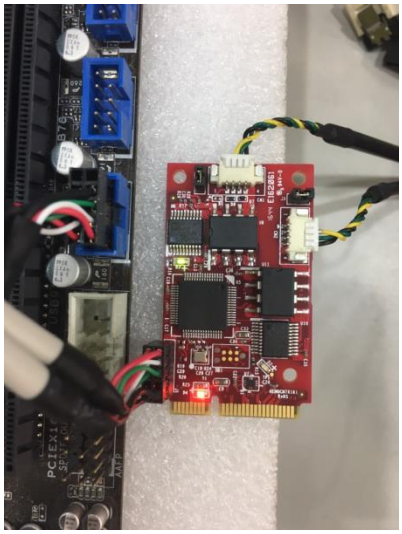
EMUC-B202 CANbus module uses USB 2.0 input interface, there are dual options to install the module.

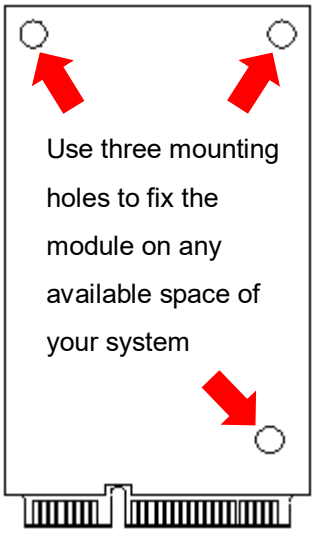
1.1.1. mPCIe Slot

Install the module to mPCIe slot which has USB 2.0 interface.

1.1.2. USB Pin Header

Don’t need to connect mPCIe golden finger, it can be connected through USB pin headers on the PCB to the motherboard. Then use three mounting holes to fix the module on any available space of your system.





NOTE: *This USB cable in the picture is not included in the package; you need to design your own USB cable.*

1.2. EGPC-B201

Install the module to M.2 B-M key slot which has PCIe interface.

2. Linux OS

Type command "lsusb" to check USB CAN device exist.

- EMUC-B202, EGPC-B201

```
jeff@inno-2034-dev:~$ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 006: ID 05e3:0608 Genesys Logic, Inc. Hub
Bus 001 Device 116: ID 413c:250e Dell Computer Corp. Dell Laser Mouse MS3220
Bus 001 Device 115: ID 1b1c:1b4f Corsair CORSAIR K68 RGB Mechanical Gaming Keyboard
Bus 001 Device 114: ID 1a40:0101 Terminus Technology Inc. Hub
Bus 001 Device 117: ID 04d8:0205 Microchip Technology, Inc. innodisk USB Dual CAN
Bus 001 Device 009: ID 0e8d:0608 MediaTek Inc. Wireless_Device
Bus 001 Device 008: ID 048d:5702 Integrated Technology Express, Inc. ITE Device
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
jeff@inno-2034-dev:~$
```

- ExUC-B2S3

```
jeff@inno-2034-dev:~$ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 006: ID 05e3:0608 Genesys Logic, Inc. Hub
Bus 001 Device 112: ID 413c:250e Dell Computer Corp. Dell Laser Mouse MS3220
Bus 001 Device 111: ID 1b1c:1b4f Corsair CORSAIR K68 RGB Mechanical Gaming Keyboard
Bus 001 Device 110: ID 1a40:0101 Terminus Technology Inc. Hub
Bus 001 Device 113: ID 196d:b003 innodisk innodisk USB Dual CAN
Bus 001 Device 009: ID 0e8d:0608 MediaTek Inc. Wireless_Device
Bus 001 Device 008: ID 048d:5702 Integrated Technology Express, Inc. ITE Device
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
jeff@inno-2034-dev:~$
```

2.1. Driver Installation

The device will be recognized as ttyACM% (%=0, 1…) by using CDC-ACM kernel driver.

Note: *Linux kernel 2.6 and above have native CDC-ACM kernel driver. Some Linux OS may need to add CDC-ACM configuration manually in building process. In different Linux OS may have different tty name.*

Type command "dmesg" to see messages below.

Generally the name would be ttyACM0 or ttyACM1 in Linux.

- EMUC-B202, EGPC-B201

```
[773300.365460] usb 1-2: USB disconnect, device number 118
[773313.427684] usb 1-2: new full-speed USB device number 122 using xhci_hcd
[773313.578166] usb 1-2: New USB device found, idVendor=04d8, idProduct=0205, bcdDevice= 1.00
[773313.578178] usb 1-2: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[773313.578183] usb 1-2: Product: innodisk USB Dual CAN
[773313.578187] usb 1-2: Manufacturer: Microchip Technology Inc.
[773313.581066] cdc_acm 1-2:1.0: ttyACM0: USB ACM device
```

- ExUC-B2S3

```
[771466.106987] usb 1-2: USB disconnect, device number 109
[771490.878368] usb 1-2: new high-speed USB device number 113 using xhci_hcd
[771491.027435] usb 1-2: New USB device found, idVendor=196d, idProduct=b003, bcdDevice= 3.00
[771491.027448] usb 1-2: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[771491.027453] usb 1-2: Product: innodisk USB Dual CAN
[771491.027457] usb 1-2: Manufacturer: innodisk
[771491.030225] cdc_acm 1-2:1.0: ttyACM0: USB ACM device
```

2.2. SocketCAN

ExUC USB CAN can support SocketCAN by additional driver and user space tool on Linux kernel 2.6.38 and above.

Before installing SocketCAN driver, you must confirm that the Linux Kernel include SocketCAN kernel module and recognize EMUC-B202 as ttyACM%(%=0,1,…) by using native CDC-ACM driver.

2.2.1. Build driver and user-space tool

Please copy kernel development packages into your system and type "make" command in root folder of this package.

There should be two output files:

- **emuc2socketcan.ko**: Kernel driver of EMUC SocketCAN
- **emucd_32** or **emucd_64**: User-space tool for enabling EMUC SocketCAN

```
root@innodisk:/home/innodisk/SocketCAN# make
make[1]: Entering directory `/home/innodisk/SocketCAN/driver'
make -C/lib/modules/`uname -r`/build M=/home/innodisk/SocketCAN/driver modules
make[2]: Entering directory `/usr/src/linux-headers-3.13.11.8-custom'
  CC [M]  /home/innodisk/SocketCAN/driver/main.o
  CC [M]  /home/innodisk/SocketCAN/driver/emuc_parse.o
  CC [M]  /home/innodisk/SocketCAN/driver/transceive.o
  LD [M]  /home/innodisk/SocketCAN/driver/emuc2socketcan.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/innodisk/SocketCAN/driver/emuc2socketcan.mod.o
  LD [M]  /home/innodisk/SocketCAN/driver/emuc2socketcan.ko
make[2]: Leaving directory `/usr/src/linux-headers-3.13.11.8-custom'
make[1]: Leaving directory `/home/innodisk/SocketCAN/driver'
make[1]: Entering directory `/home/innodisk/SocketCAN/utility'
  Compiling 'main.c' ...
  Building 'emucd_64' VER=...
make[1]: Leaving directory `/home/innodisk/SocketCAN/utility'
root@innodisk:/home/innodisk/SocketCAN#
```

You can type “emucd_64 -h” for help.

```
inno@inno-pc:~/svn/Trunk/EP/EMUC_B202/Linux/SocketCAN/utility$ ./emucd_64

Usage: ./emucd_64 [options] <tty> [canif-name] [canif2-name]

Options: -s <speed>[<speed>] (set CAN speed 3..7)
                4: 100  KBPS
                5: 125  KBPS
                6: 250  KBPS
                7: 500  KBPS
                8: 800  KBPS
                9: 1000 KBPS
                A: 400  KBPS
         -e <errorType>[<errorType>] (set CANbus error type)
                0: EMUC_DIS_ALL
                1: EMUC_EE_ERR
                2: EMUC_BUS_ERR
                3: EMUC_EN_ALL
         -F          (stay in foreground; no daemonize)
         -h          (show this help page)
         -v          (show version info)
         -t          (set open tty device timeout [sec])

Examples:
emucd_64 -v /dev/ttyACM0
emucd_64 -s7 /dev/ttyACM0
emucd_64 -s7 -e3 /dev/ttyACM0
emucd_64 -s79 /dev/ttyACM0 can0 can1
emucd_64 -s79 -t10 /dev/ttyACM0 can0 can1
(Note: emucd_32 for 32-bit OS)
```

./emucd_64 -s7 /dev/ttyACM0 (500 KBPS on both channel)
./emucd_64 -s79 /dev/ttyACM0 (500 KBPS on ch1, 1000 KBPS on ch2)

NOTE: *If you don’t specify interface name, default name will be “emuccan0” and “emuccan1”*

2.2.2. SocketCAN Driver Installation

There are shell scripts “start.sh” and “end.sh” to install the driver and enable SocketCAN interface.

start.sh

Please modify the baud rate and tty port setting depend on the environment needs.

```
### parameter
socket_name_1=emuccan0
socket_name_2=emuccan1
dev_name=ttyACM0
baudrate=7  # 0~3: support FW version >= 03.00
            # 0: 5   KBPS, 1: 10  KBPS,  2: 20  KBPS, 3: 50  KBPS,
            # 4: 100 KBPS, 5: 125 KBPS,  6: 250 KBPS, 7: 500 KBPS,
            # 8: 800 KBPS, 9: 1 MBPS,   10: 400 KBPS
error_type=0 # 0: EMUC_DIS_ALL, 1: EMUC_EE_ERR, 2: EMUC_BUS_ERR, 3: EMUC_EN_ALL
```

NOTE: Only ExUC-BxS3 supports baud rate 5/10/20/50k

end.sh

```
sudo pkill -2 emucd_64
sleep 0.2
sudo rmmod emuc2socketcan
#rm /lib/modules/$(uname -r)/kernel/drivers/net/can/emuc2socketcan.ko
```

You can start/end SocketCAN interface simply by using the scripts.

-$ chmod +x start.sh

-$ ./start.sh

You can see the CAN interface name by "ifconfig" command.

```
root@innodisk:/home/innodisk/SocketCAN# ifconfig
can0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          UP RUNNING NOARP  MTU:16  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:10
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

can1      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          UP RUNNING NOARP  MTU:16  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:10
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Base address:0x101
```

2.2.3. CAN-utils

After SocketCAN setup is finished, you can use open source project "can-utils" to test by "cansend" and "candump".

(https://github.com/linux-can/can-utils).

- Install CAN-utils

- $ apt-get install can-utils

- use can0 to send and can1 to receive.

```
yichen@yichen-MS-7971:~$ cansend can0 111#1122334455667788
yichen@yichen-MS-7971:~$ cansend can0 111#1122334455667788
yichen@yichen-MS-7971:~$ cansend can0 111#1122334455667788
yichen@yichen-MS-7971:~$ cansend can0 111#R1
yichen@yichen-MS-7971:~$ cansend can0 111#R2
yichen@yichen-MS-7971:~$ cansend can0 111#R3
yichen@yichen-MS-7971:~$
```

```
yichen@yichen-MS-7971:~$ candump can1
  can1  111   [8]  11 22 33 44 55 66 77 88
  can1  111   [8]  11 22 33 44 55 66 77 88
  can1  111   [8]  11 22 33 44 55 66 77 88
  can1  111   [1]  remote request
  can1  111   [2]  remote request
  can1  111   [3]  remote request
```

2.2.4. Boot Up Script

We provide Linux boot up script to initial SocketCAN interface automatically after system boot up.

run_emucd

Please modify the baud rate and tty port setting depend on the environment needs.

```
### parameter
socket_name_1=can0
socket_name_2=can1
dev_name=ttyACM0
baudrate=7  # 0~3: support FW version >= 03.00
            # 0: 5   KBPS, 1: 10  KBPS,  2: 20  KBPS, 3: 50  KBPS,
            # 4: 100 KBPS, 5: 125 KBPS,  6: 250 KBPS, 7: 500 KBPS,
            # 8: 800 KBPS, 9: 1 MBPS,   10: 400 KBPS
error_type=0 # 0: EMUC_DIS_ALL, 1: EMUC_EE_ERR, 2: EMUC_BUS_ERR, 3: EMUC_EN_ALL
```

NOTE: Only ExUC-BxS3 supports baud rate 5/10/20/50k

Run the following command in the “release” folder to add/remove boot up script.

- $ chmod +x add_2_boot.sh
- $ ./add_2_boot.sh

```
yichen@yichen-MS-7971:~/svn/Inno/Trunk/EP/EMUC_B202/Linux/SocketCAN/bootexec$ ./add_2_boot.sh
yichen@yichen-MS-7971:~/svn/Inno/Trunk/EP/EMUC_B202/Linux/SocketCAN/bootexec$
```

- $ chmod +x remove_boot.sh
- $ ./ remove_boot.sh

```
ichen@yichen-MS-7971:~/svn/Inno/Trunk/EP/EMUC_B202/Linux/SocketCAN/bootexec$ ./remove_boot.sh
ichen@yichen-MS-7971:~/svn/Inno/Trunk/EP/EMUC_B202/Linux/SocketCAN/bootexec$
```

2.2.5. CAN Error Frame

CAN error frame can be dumped by adding the parameter “-e” when running the emucd_32 or emucd_64 utility.

```
emucd_64 -s7 -e3 /dev/ttyACM0
```

It can be simply set the error type by editing “start.sh”.
“run_emucd” of boot up script has this parameter as well.

```
### parameter
socket_name_1=can0
socket_name_2=can1
dev_name=ttyACM0
baudrate=7 # 4: 100 KBPS, 5: 125 KBPS,  6: 250 KBPS, 7: 500 KBPS,
           # 8: 800 KBPS, 9: 1 MBPS,   10: 400 KBPS
error_type=0 # 0: EMUC_DIS_ALL, 1: EMUC_EE_ERR, 2: EMUC_BUS_ERR, 3: EMUC_EN_ALL
```

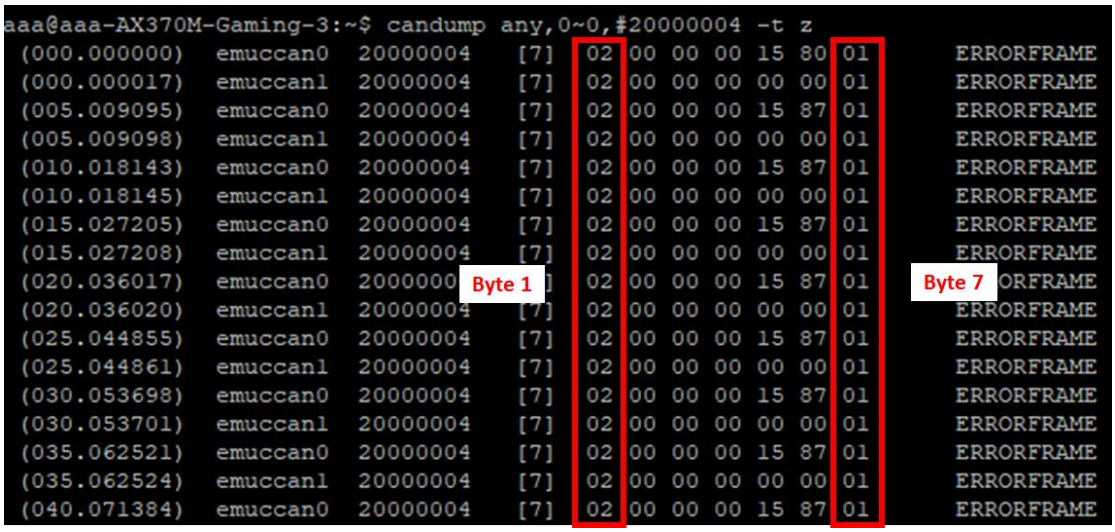
0: EMUC_DIS_ALL: disable all error frame.

1: EMUC_EE_ERR: enable EEPROM error only.

2: EMUC_BUS_ERR: enable CAN bus error only.

3: EMUC_EM_ALL: enable both EERPOM and CAN bus error.

CAN error frame can be dumped through the following command of CAN-utils.



Byte1: Error Type, 0x01=EEPROM Error, 0x02=Bus Error

Byte2~Byte7: Bus Error Register, please refer to *3.2.Register mapping table of CAN error status.*

3. Appendix

3.1. Example of CAN acceptance filter

The filter mask is used to determine which bits in the identifier of the received f are compared with the filter

- If a mask bit is set to a zero, the corresponding ID bit will automatically be accepted, regardless of the value of the filter bit.

- If a mask bit is set to a one, the corresponding ID bit will be compare with the value of the filter bit; if they match it is accepted otherwise the frames is rejected.

Example 1:

We wish to accept only frames with ID of 00001567 (hexadecimal values)

- set filter to 00001567
- set mask to 1FFFFFFF

When a frame arrives its ID is compared with the filter and all bits must match; any frame that does not match ID 00001567 is rejected

Example 2:

We wish to accept only frames with IDs of 00001560 through to 0000156F

- set filter to 00001560
- set mask to 1FFFFFF0

When a frame arrives its ID is compared with the filter and all bits except bits 0 to 3 must match; any other frame is rejected

Example 3:

We wish to accept only frames with IDs of 00001560 through to 00001567

- set filter to 00001560
- set mask to 1FFFFFF8

When a frame arrives its ID is compared with the filter and all bits except bits 0 to 2 must match; any other frame is rejected

Example 4:

We wish to accept any frame

- set filter to 0
- set mask to 0

All frames are accepted

3.2. Register mapping table of CAN error status

bit 21 TXBO: Transmitter in Error State Bus OFF (TERRCNT >= 256)

bit 20 TXBP: Transmitter in Error State Bus Passive (TERRCNT >= 128)

bit 19 RXBP: Receiver in Error State Bus Passive (RERRCNT >= 128)

bit 18 TXWARN: Transmitter in Error State Warning (128 > TERRCNT >= 96)

bit 17 RXWARN: Receiver in Error State Warning (128 > RERRCNT >= 96)

bit 16 EWARN: Transmitter or Receiver is in Error State Warning

bit 15-8 TERRCNT<7:0>: Transmit Error Counter

bit 7-0 RERRCNT<7:0>: Receive Error Counter

Contact us

Headquarters (Taiwan)

5F., No. 237, Sec. 1, Datong Rd., Xizhi Dist., New Taipei City 221, Taiwan

Tel: +886-2-77033000

Email: sales@innodisk.com

Branch Offices:

USA

usasales@innodisk.com

+1-510-770-9421

Europe

eusales@innodisk.com

+31-40-3045-400

Japan

jpsales@innodisk.com

+81-3-6667-0161

China

sales_cn@innodisk.com

+86-755-21673689

www.innodisk.com

March 18, 2024