innodisk

EMUC-B201

USB to CANbus

User Manual

Rev 1.5

Revision History

Revision	Date	Description
1.0	2015/11/06	Initial Release
1.1	2016/06/21	1. Revise com port parameter description in section 3. 2. Add Linux com description in section 3.
1.2	2016/07/20	Add section 2.3 SocketCAN
1.3	2016/10/4	Add section 3.1 COM Port Selection
1.4	2016/10/31	1. Remove 3.2 to 3.7 that can find in lib_emuc.h. 2. Add new function EMUCReceiveNonblock() in section 3.2 Function Description.
1.5	2016/12/15	1. Update Linux COM table in 3.1 COM Port Selection.

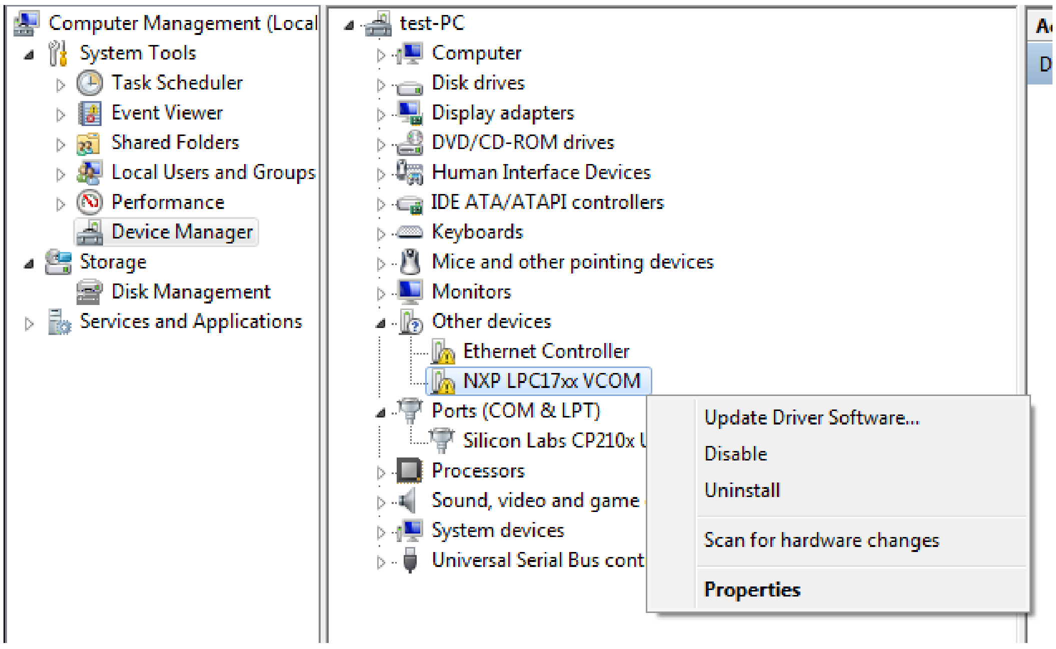
Table of Contents

1. Windows

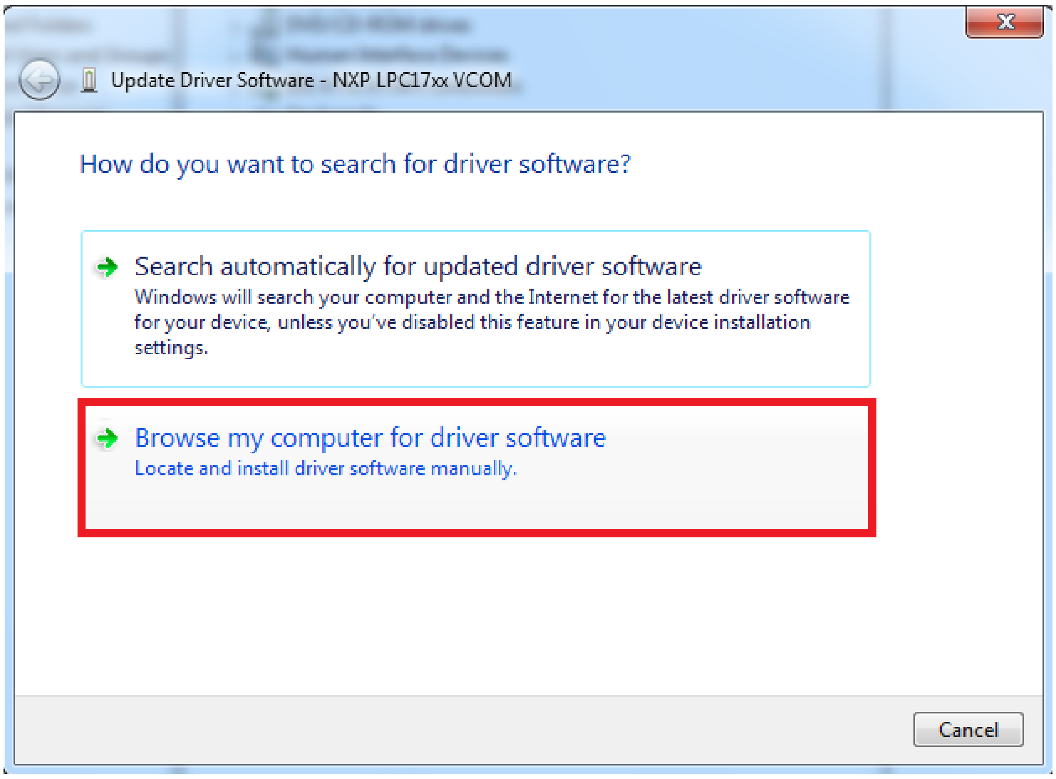
1.1. Install Driver

Install EMUC-B201 either into mPCIe slot or with USB pin header. The device named "NXP LPC 17xx VCOM" can be found in "Device Manager".

Click "Update Driver Software" to install driver.



Select "Browse my computer for driver software"


Update Driver Software - NXP LPC17xx VCOM

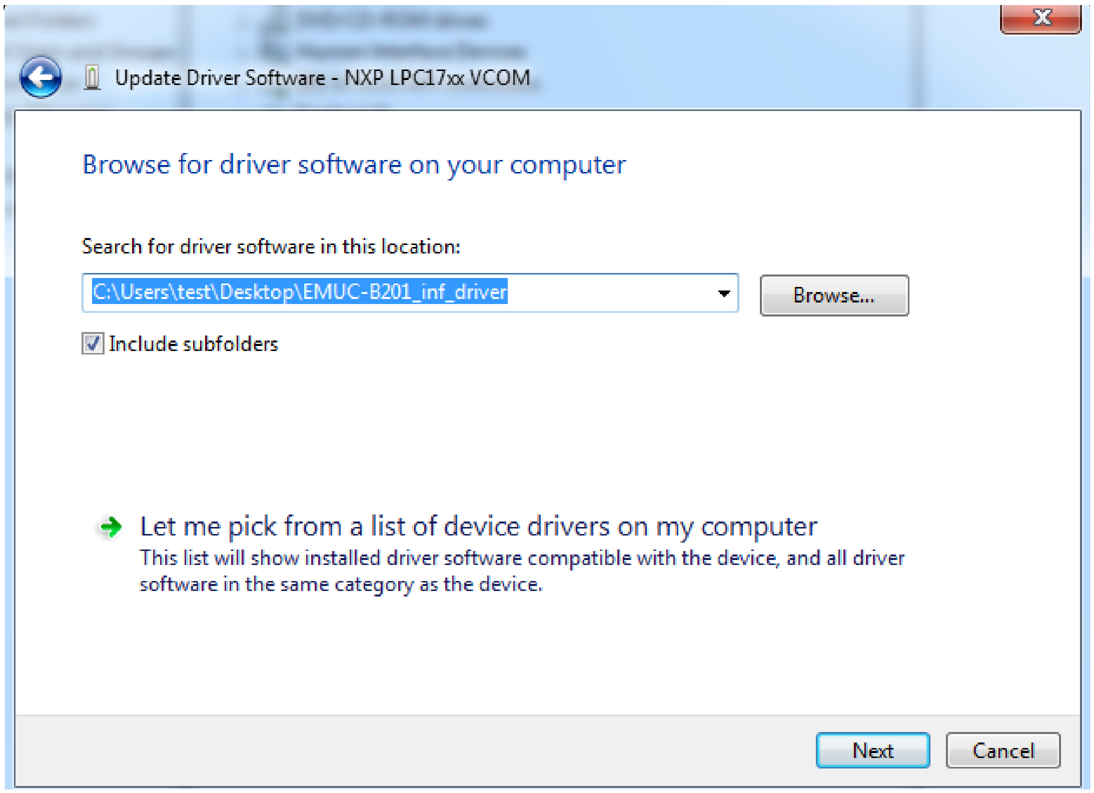
How do you want to search for driver software?

Search automatically for updated driver software
Windows will search your computer and the Internet for the latest driver software for your device, unless you've disabled this feature in your device installation settings.

Browse my computer for driver software
Locate and install driver software manually.

Cancel

Browse the path linking to EMUC-B201 driver.



Update Driver Software - NXP LPC17xx VCOM

Browse for driver software on your computer

Search for driver software in this location:

C:\Users\test\Desktop\EMUC-B201_inf_driver

Browse...

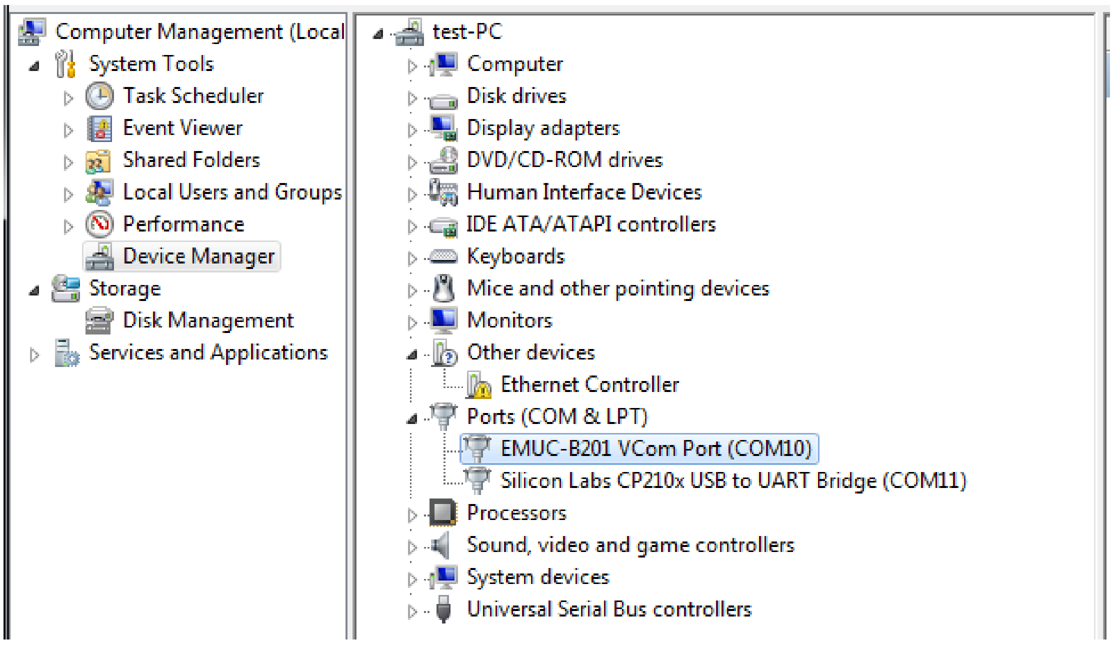
☑ Include subfolders

Let me pick from a list of device drivers on my computer

This list will show installed driver software compatible with the device, and all driver software in the same category as the device.

Next Cancel

After installing the driver, device can be recognized as a COM port named "EMUC-B201 VCom Port".

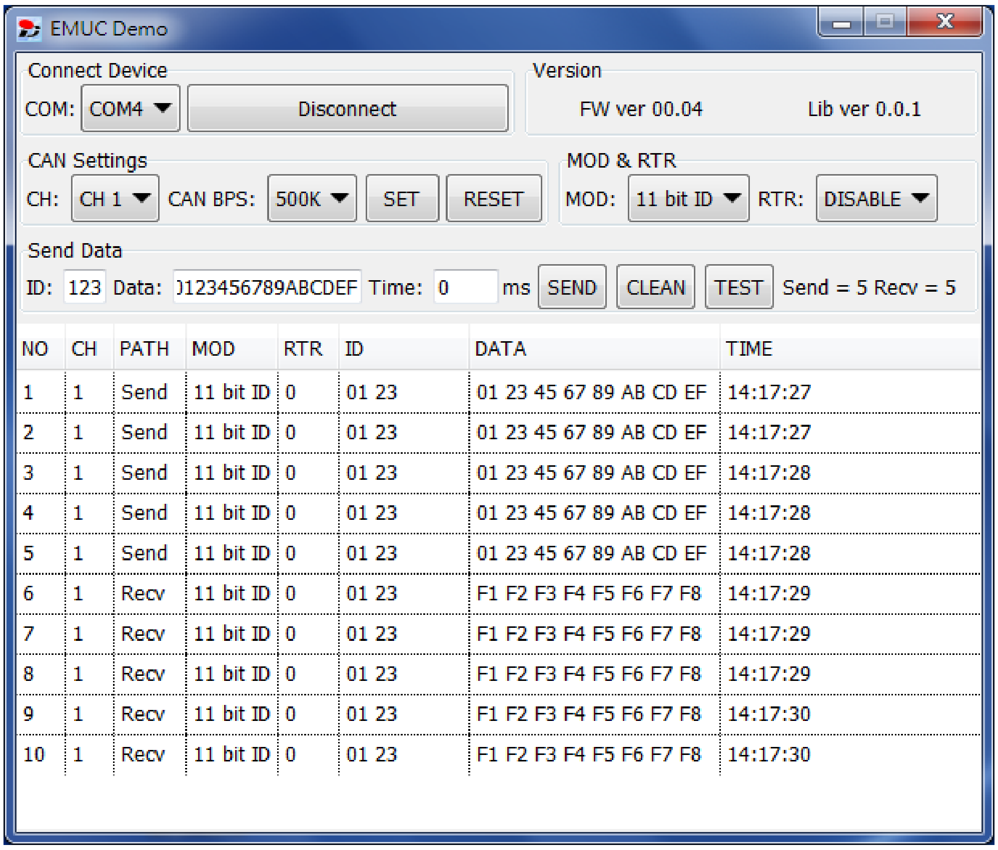


Computer Management (Local
- System Tools
  - Task Scheduler
  - Event Viewer
  - Shared Folders
  - Local Users and Groups
  - Performance
  - Device Manager
- Storage
  - Disk Management
- Services and Applications

test-PC
- Computer
- Disk drives
- Display adapters
- DVD/CD-ROM drives
- Human Interface Devices
- IDE ATA/ATAPI controllers
- Keyboards
- Mice and other pointing devices
- Monitors
- Other devices
  - Ethernet Controller
- Ports (COM & LPT)
  - EMUC-B201 VCom Port (COM10)
  - Silicon Labs CP210x USB to UART Bridge (COM11)
- Processors
- Sound, video and game controllers
- System devices
- Universal Serial Bus controllers

1.2. EMUC-B201 Test Utility

You can use this tool to test EMUC-B201.



EMUC Demo

Connect Device — COM: COM4 — Disconnect

Version — FW ver 00.04 — Lib ver 0.0.1

CAN Settings — CH: CH 1 — CAN BPS: 500K — SET — RESET

MOD & RTR — MOD: 11 bit ID — RTR: DISABLE

Send Data — ID: 123 — Data:)123456789ABCDEF — Time: 0 ms — SEND — CLEAN — TEST — Send = 5 Recv = 5

NO	CH	PATH	MOD	RTR	ID	DATA	TIME
1	1	Send	11 bit ID	0	01 23	01 23 45 67 89 AB CD EF	14:17:27
2	1	Send	11 bit ID	0	01 23	01 23 45 67 89 AB CD EF	14:17:27
3	1	Send	11 bit ID	0	01 23	01 23 45 67 89 AB CD EF	14:17:28
4	1	Send	11 bit ID	0	01 23	01 23 45 67 89 AB CD EF	14:17:28
5	1	Send	11 bit ID	0	01 23	01 23 45 67 89 AB CD EF	14:17:28
6	1	Recv	11 bit ID	0	01 23	F1 F2 F3 F4 F5 F6 F7 F8	14:17:29
7	1	Recv	11 bit ID	0	01 23	F1 F2 F3 F4 F5 F6 F7 F8	14:17:29
8	1	Recv	11 bit ID	0	01 23	F1 F2 F3 F4 F5 F6 F7 F8	14:17:29
9	1	Recv	11 bit ID	0	01 23	F1 F2 F3 F4 F5 F6 F7 F8	14:17:30
10	1	Recv	11 bit ID	0	01 23	F1 F2 F3 F4 F5 F6 F7 F8	14:17:30

Connect Device

COM: Select the port which is recognized as “EMUC VCom Port” in Device Manager, then click “Connect”

CAN Setting

CH: Send frame from CAN1, CAN2 or both.

CAN BPS: Set CAN baudrate, EMUC supports 50K, 125K, 250K, 500K, 1000K, after selecting please click “SET” to take effect.

RESET: Reset CAN port to clear register.

MOD & RTR

MOD: Select 11 or 29 bit CAN ID mode.

RTR: Enable or disable Remote Transmit Request.

Send Data

ID: Input CAN ID, maximum of 11bit and 29bit is 7FF and 1FFFFFFF.

Data: Input data , max is 8 byte.

Time: Set time interval of sending frame. It will only send once when time =0. The Minimum value is 10ms. During sending, you can click “SEND” again or set time=0 to stop it.

CLEAN: Clear the record list.

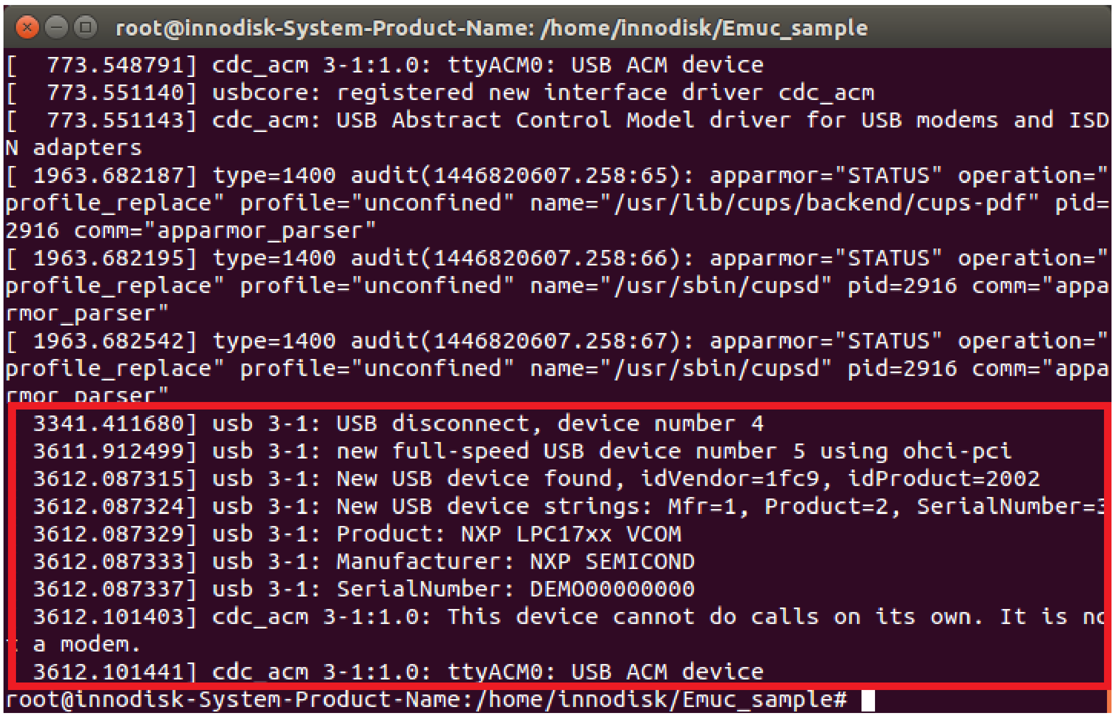
2. Linux

2.1. Install Driver

Install EMUC-B201 either into mPCIe slot or with USB pin header. The device will be recognized as ttyACM% (%=0,1…) by using native CDC-ACM driver.

Type command “dmesg” to see messages below.

Generally the name would be ttyACM0 or ttyACM1 in Linux.



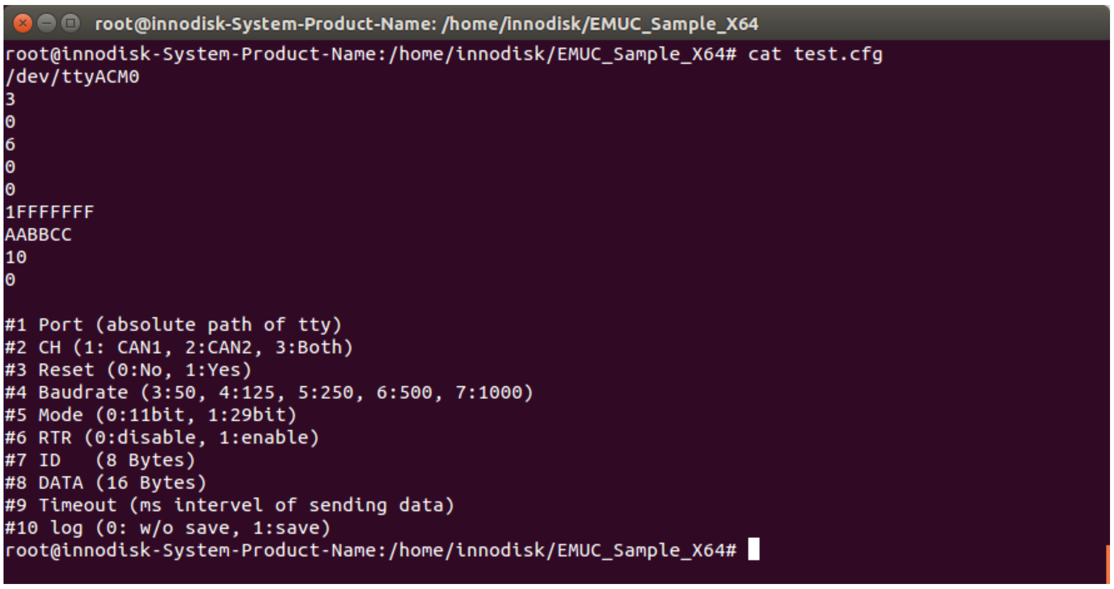
root@innodisk-System-Product-Name: /home/innodisk/Emuc_sample

```
[  773.548791] cdc_acm 3-1:1.0: ttyACM0: USB ACM device
[  773.551140] usbcore: registered new interface driver cdc_acm
[  773.551143] cdc_acm: USB Abstract Control Model driver for USB modems and ISD
N adapters
[ 1963.682187] type=1400 audit(1446820607.258:65): apparmor="STATUS" operation="
profile_replace" profile="unconfined" name="/usr/lib/cups/backend/cups-pdf" pid=
2916 comm="apparmor_parser"
[ 1963.682195] type=1400 audit(1446820607.258:66): apparmor="STATUS" operation="
profile_replace" profile="unconfined" name="/usr/sbin/cupsd" pid=2916 comm="appa
rmor_parser"
[ 1963.682542] type=1400 audit(1446820607.258:67): apparmor="STATUS" operation="
profile_replace" profile="unconfined" name="/usr/sbin/cupsd" pid=2916 comm="appa
rmor_parser"
  3341.411680] usb 3-1: USB disconnect, device number 4
  3611.912499] usb 3-1: new full-speed USB device number 5 using ohci-pci
  3612.087315] usb 3-1: New USB device found, idVendor=1fc9, idProduct=2002
  3612.087324] usb 3-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
  3612.087329] usb 3-1: Product: NXP LPC17xx VCOM
  3612.087333] usb 3-1: Manufacturer: NXP SEMICOND
  3612.087337] usb 3-1: SerialNumber: DEMO00000000
  3612.101403] cdc_acm 3-1:1.0: This device cannot do calls on its own. It is no
  a modem.
  3612.101441] cdc_acm 3-1:1.0: ttyACM0: USB ACM device
root@innodisk-System-Product-Name:/home/innodisk/Emuc_sample# 
```

2.2. EMUC-B201 Test Utility

You can use “EMUC_Sample” to test EMUC-B201.

Before executing the command, you must check test.cfg below.

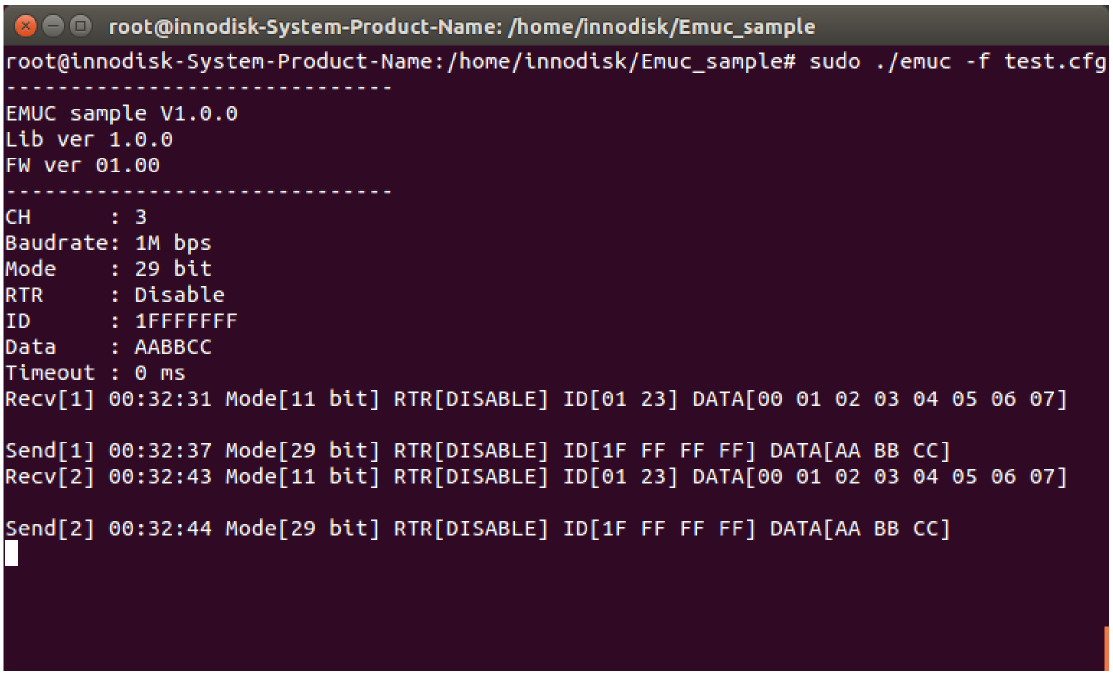


root@innodisk-System-Product-Name: /home/innodisk/EMUC_Sample_X64

```
root@innodisk-System-Product-Name:/home/innodisk/EMUC_Sample_X64# cat test.cfg
/dev/ttyACM0
3
0
6
0
0
1FFFFFFF
AABBCC
10
0

#1 Port (absolute path of tty)
#2 CH (1: CAN1, 2:CAN2, 3:Both)
#3 Reset (0:No, 1:Yes)
#4 Baudrate (3:50, 4:125, 5:250, 6:500, 7:1000)
#5 Mode (0:11bit, 1:29bit)
#6 RTR (0:disable, 1:enable)
#7 ID   (8 Bytes)
#8 DATA (16 Bytes)
#9 Timeout (ms intervel of sending data)
#10 log (0: w/o save, 1:save)
root@innodisk-System-Product-Name:/home/innodisk/EMUC_Sample_X64# 
```

Execute "sudo ./emuc -f test.cfg" to start thread by root privilege.

When Timeout=0ms, you can press "Enter" to send frame once.



```
root@innodisk-System-Product-Name: /home/innodisk/Emuc_sample
root@innodisk-System-Product-Name:/home/innodisk/Emuc_sample# sudo ./emuc -f test.cfg
------------------------------
EMUC sample V1.0.0
Lib ver 1.0.0
FW ver 01.00
------------------------------
CH       : 3
Baudrate: 1M bps
Mode     : 29 bit
RTR      : Disable
ID       : 1FFFFFFF
Data     : AABBCC
Timeout  : 0 ms
Recv[1] 00:32:31 Mode[11 bit] RTR[DISABLE] ID[01 23] DATA[00 01 02 03 04 05 06 07]

Send[1] 00:32:37 Mode[29 bit] RTR[DISABLE] ID[1F FF FF FF] DATA[AA BB CC]
Recv[2] 00:32:43 Mode[11 bit] RTR[DISABLE] ID[01 23] DATA[00 01 02 03 04 05 06 07]

Send[2] 00:32:44 Mode[29 bit] RTR[DISABLE] ID[1F FF FF FF] DATA[AA BB CC]
```

2.3. SocketCAN

EMUC-B201 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-B201 as ttyACM%(%=0,1,…) by using native CDC-ACM driver.

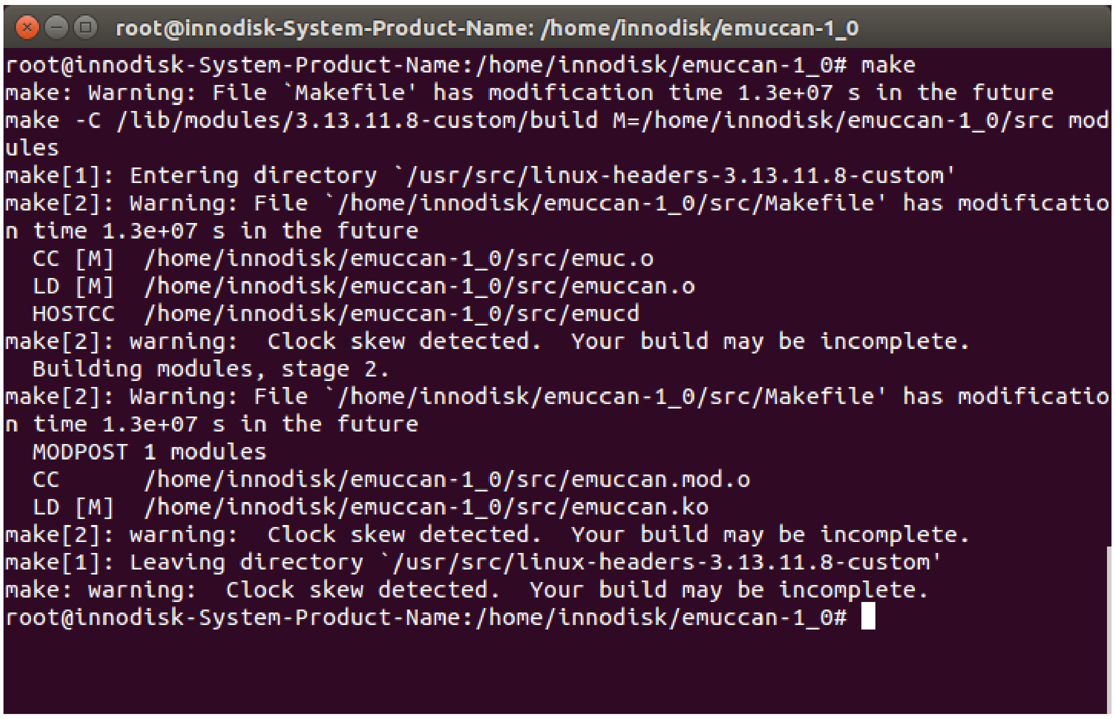
The following example uses Ubuntu 14.04.

2.3.1. 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:

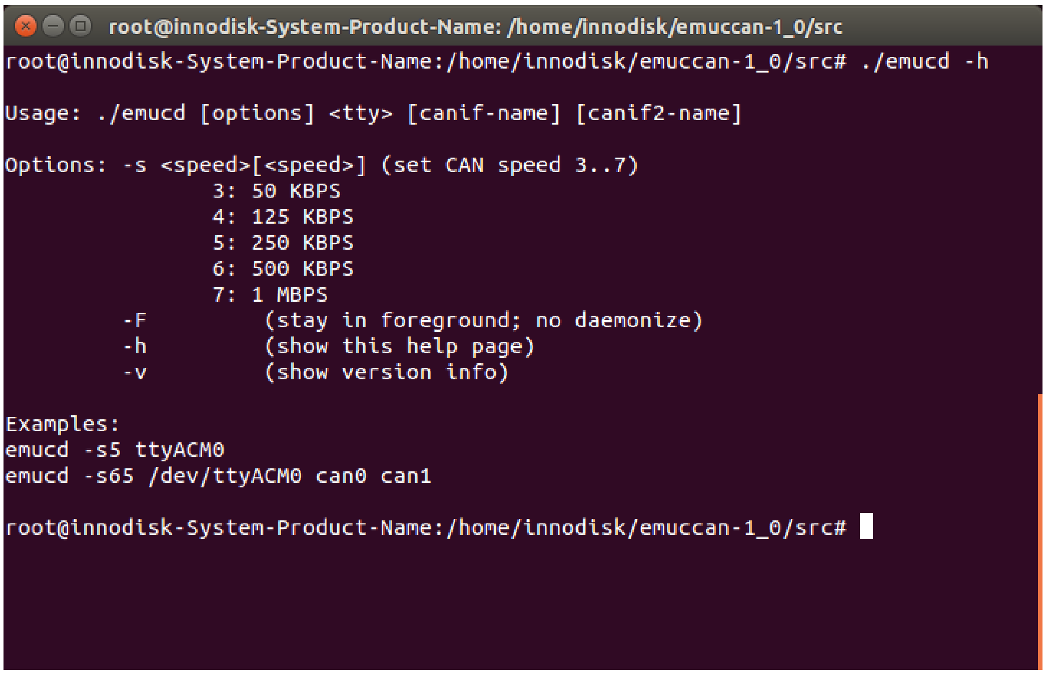
- src/emuccan.ko : Kernel driver of EMUC SocketCAN
- src/emucd :User-space tool for enabling EMUC SocketCAN


root@innodisk-System-Product-Name: /home/innodisk/emuccan-1_0

```
root@innodisk-System-Product-Name:/home/innodisk/emuccan-1_0# make
make: Warning: File `Makefile' has modification time 1.3e+07 s in the future
make -C /lib/modules/3.13.11.8-custom/build M=/home/innodisk/emuccan-1_0/src mod
ules
make[1]: Entering directory `/usr/src/linux-headers-3.13.11.8-custom'
make[2]: Warning: File `/home/innodisk/emuccan-1_0/src/Makefile' has modificatio
n time 1.3e+07 s in the future
  CC [M]  /home/innodisk/emuccan-1_0/src/emuc.o
  LD [M]  /home/innodisk/emuccan-1_0/src/emuccan.o
  HOSTCC  /home/innodisk/emuccan-1_0/src/emucd
make[2]: warning:  Clock skew detected.  Your build may be incomplete.
  Building modules, stage 2.
make[2]: Warning: File `/home/innodisk/emuccan-1_0/src/Makefile' has modificatio
n time 1.3e+07 s in the future
  MODPOST 1 modules
  CC      /home/innodisk/emuccan-1_0/src/emuccan.mod.o
  LD [M]  /home/innodisk/emuccan-1_0/src/emuccan.ko
make[2]: warning:  Clock skew detected.  Your build may be incomplete.
make[1]: Leaving directory `/usr/src/linux-headers-3.13.11.8-custom'
make: warning:  Clock skew detected.  Your build may be incomplete.
root@innodisk-System-Product-Name:/home/innodisk/emuccan-1_0#
```

2.3.2. Usage and Example

After installing driver by "insmod" command, you can set CAN speed for two channels by executing "emucd" daemonYou can type "emucd -h" for help.


root@innodisk-System-Product-Name: /home/innodisk/emuccan-1_0/src

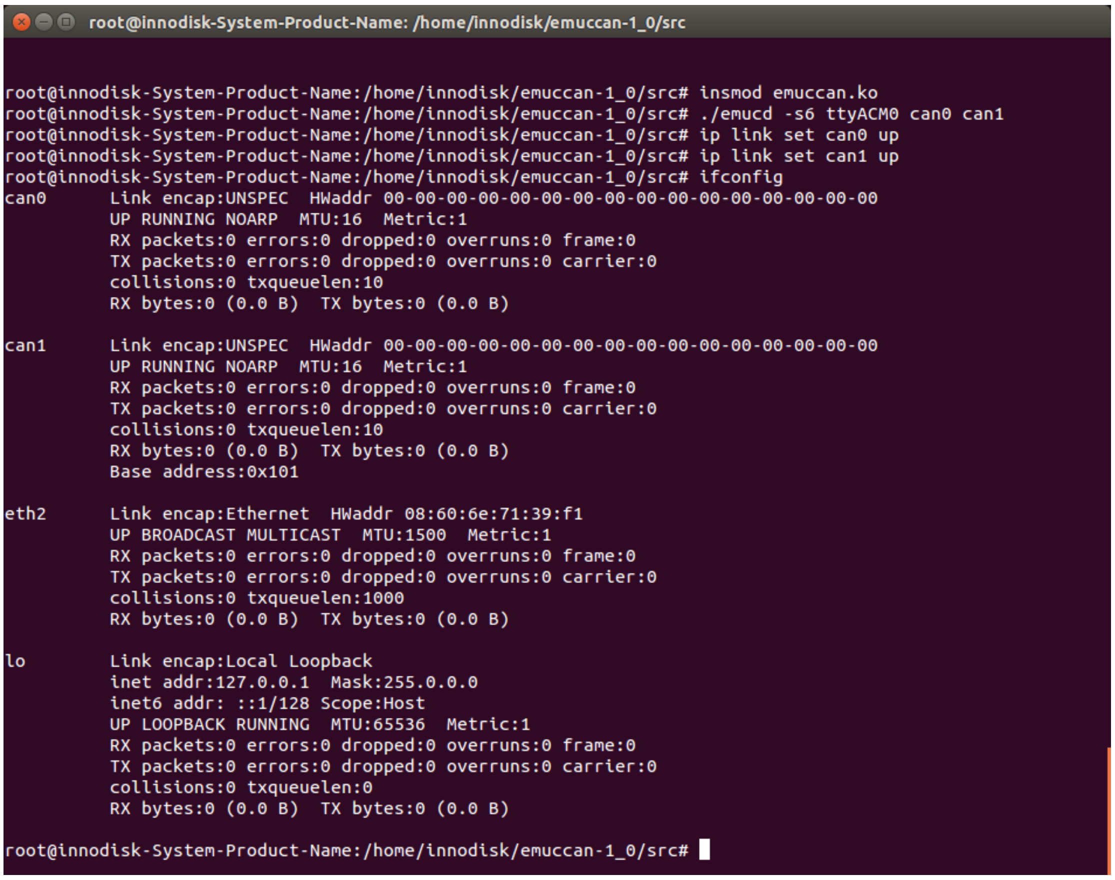
```
root@innodisk-System-Product-Name:/home/innodisk/emuccan-1_0/src# ./emucd -h

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

Options: -s <speed>[<speed>] (set CAN speed 3..7)
                3: 50 KBPS
                4: 125 KBPS
                5: 250 KBPS
                6: 500 KBPS
                7: 1 MBPS
         -F         (stay in foreground; no daemonize)
         -h         (show this help page)
         -v         (show version info)

Examples:
emucd -s5 ttyACM0
emucd -s65 /dev/ttyACM0 can0 can1

root@innodisk-System-Product-Name:/home/innodisk/emuccan-1_0/src#
```

./emucd -s6 /dev/ttyACM0   (500 KBPS on both channel)

./emucd -s34 /dev/ttyACM0 (50 KBPS on ch1, 125 KBPS on ch2)

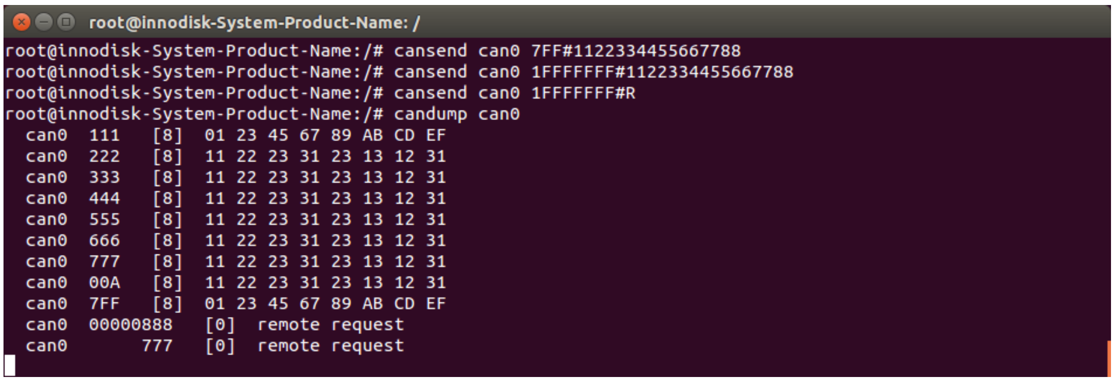
NOTICE: If you don't specify interface name, default name will be "emuccan0" and "emuccan1"

The picture below is an example to set EMUC to network interface.
You can see the CAN interface name by "ifconfig" command.



```
root@innodisk-System-Product-Name: /home/innodisk/emuccan-1_0/src

root@innodisk-System-Product-Name:/home/innodisk/emuccan-1_0/src# insmod emuccan.ko
root@innodisk-System-Product-Name:/home/innodisk/emuccan-1_0/src# ./emucd -s6 ttyACM0 can0 can1
root@innodisk-System-Product-Name:/home/innodisk/emuccan-1_0/src# ip link set can0 up
root@innodisk-System-Product-Name:/home/innodisk/emuccan-1_0/src# ip link set can1 up
root@innodisk-System-Product-Name:/home/innodisk/emuccan-1_0/src# 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

eth2      Link encap:Ethernet  HWaddr 08:60:6e:71:39:f1
          UP BROADCAST MULTICAST  MTU:1500  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:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  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:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@innodisk-System-Product-Name:/home/innodisk/emuccan-1_0/src# 
```

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).



```
root@innodisk-System-Product-Name: /
root@innodisk-System-Product-Name:/# cansend can0 7FF#1122334455667788
root@innodisk-System-Product-Name:/# cansend can0 1FFFFFFF#1122334455667788
root@innodisk-System-Product-Name:/# cansend can0 1FFFFFFF#R
root@innodisk-System-Product-Name:/# candump can0
  can0  111   [8]  01 23 45 67 89 AB CD EF
  can0  222   [8]  11 22 23 31 23 13 12 31
  can0  333   [8]  11 22 23 31 23 13 12 31
  can0  444   [8]  11 22 23 31 23 13 12 31
  can0  555   [8]  11 22 23 31 23 13 12 31
  can0  666   [8]  11 22 23 31 23 13 12 31
  can0  777   [8]  11 22 23 31 23 13 12 31
  can0  00A   [8]  11 22 23 31 23 13 12 31
  can0  7FF   [8]  01 23 45 67 89 AB CD EF
  can0  00000888   [0]  remote request
  can0       777   [0]  remote request
```

3. Software API

EMUC API is based on a dynamic library (DLL) in Windows and static library (.a) in Linux to control EMUC-B201.
lib_emuc.h includes declaration and data structure requested for programming.

3.1. COM Port Selection

EMUC-B201 is connected by virtual COM port using CDC-ACM driver.
COM port parameter of API must be given an "int" value instead of a real port name or port number in the OS.

Windows

Real COM port number-1 would be the "int" value for API.
Example: 0=COM1, 1=COM2, 2=COM3…254=COM255, 255=COM256

Linux

EMUC-B201 supports the following COM names. The port mapping to below "int" value start from 0. Generally the name would be ttyACM0 or ttyACM1 in Linux.
Example: 24=ttyACM0, 25=ttyACM1

Index	Port	Index	Port	Index	Port
0	ttyS0	1	ttyS1	2	ttyS2
3	ttyS3	4	ttyS4	5	ttyS5
6	ttyS6	7	ttyS7	8	ttyS8
9	ttyS9	10	ttyS10	11	ttyS11
12	ttyS12	13	ttyS13	14	ttyS14
15	ttyS15	16	ttyUSB0	17	ttyUSB1
18	ttyUSB2	19	ttyUSB3	20	ttyUSB4
21	ttyUSB5	22	ttyAMA0	23	ttyAMA1
24	ttyACM0	25	ttyACM1	26	ttyACM2
27	ttyACM3	28	ttyACM4	29	ttyACM5
30	ttyACM6	31	ttyACM7	32	ttyACM8
33	ttyACM9	34	ttyACM10	35	ttyACM11
36	ttyACM12	37	ttyACM13	38	ttyACM14
39	ttyACM15	40	rfcomm0	41	Rfcomm1
42	Ircomm0	43	Ircomm1	44	cuau0
45	cuau1	46	cuau2	47	cuau3
48	cuaU0	49	cuaU1	50	cuaU2
51	cuaU3				

3.2. Function Description

This chapter describes API functions and parameters.

3.2.1. EMUCShowVer

Description: Get firmware and library version.

SYSTAX:

```
int EMUCShowVer(int port, VER_INFO *ver)
```

VER_INFO struct:

```
typedef struct
{
    char fw[VER_LEN];
    char api[VER_LEN];
} VER_INFO;
```

Member:

fw: [output] Firmware version, length 16 bytes

api: [output] API version, length 16 bytes

Return Status Code:

Value	**Return Value**
0	SUCCESS
-1	EXCEPETION

3.2.2. EMUCOpenDevice

Description: Open virtual COM port.

SYSTAX:

```
int EMUCOpenDevice(int port)
```

Member:

port: [input] The virtual COM port number.

Return Status Code:

Value	**Return Value**
0	SUCCESS
1	COM_NOT_EXIST

5	COM_IN_USE

3.2.3. EMUCCloseDevice

Description: Close virtual COM port.

SYSTAX:

```
void EMUCCloseDevice(int port)
```

Member:

port: [input] The virtual COM port number.

3.2.4. EMUCSetCAN

Description: Set baud rate of CAN port.

SYSTAX:

```
int EMUCSetCAN(int port, int ch, int bdrate)
```

Member:

port: [input] The virtual COM port number.

ch: [input] The CAN port number. (1:CAN1, 2:CAN2, 3:Both)

CAN Port	VALUE
CAN 1	EMUC_CH1
CAN 2	EMUC_CH2
Both	EMUC_BOTH

bdrate: [input] The baud rate of the CAN port

Baud rate	VALUE
50 kbps	EMUC_BAUDRATE_50K
125 kbps	EMUC_BAUDRATE_125K
250 kbps	EMUC_BAUDRATE_250K
500 kbps	EMUC_BAUDRATE_500K
1M bps	EMUC_BAUDRATE_1M

Return Status Code:

Value	Return Value
0	SUCCESS
-1	UNSUCCESS

3.2.5. EMUCResetCAN

Description: Reset CAN port and clearing register without changing baudrate

SYSTAX:

```
int EMUCResetCAN(int port)
```

Member:

port: [input] The virtual COM port number.

Return Status Code:

Value	Return Value
0	SUCCESS
-1	RESET FAIL

3.2.6. EMUCSend

Description: Send data from USB to CAN.

SYSTAX:

```
int EMUCSend (DATA_INFO *info)
```

DATA_INFO struct:

```
typedef struct
{
        int com_port;
        int channel;
        int mod;
        int rtr;
        int dlc;
        unsigned char id[ID_LEN];
        unsigned char data[DATA_LEN];
} DATA_INFO;
```

Member:

com_port: [input] The virtual COM port number.

channel: [input] The CAN port number. (1:CAN1, 2:CAN2, 3:Both)

mod: [input] ID mode (0:11 bit, 1:29 bit)

rtr: [input] Remote transmit request (0:disable, 1:enable)

dlc: [input] Data length (Range 0 ~ 8)

id[ID_LEN]: [input] CAN ID (ID_LEN = 4)

data[DATA_LEN]: Data (DATA_LEN = 8)

3.2.7. EMUCReceive

Description: Receive one data from CAN to USB.

SYSTAX:

```
int EMUCReceive (DATA_INFO *info)
```

DATA_INFO struct:

```
typedef struct
{
        int com_port;
        int channel;
        int mod;
        int rtr;
        int dlc;
        unsigned char id[ID_LEN];
        unsigned char data[DATA_LEN];
} DATA_INFO;
```

Member:

com_port: [input] The virtual COM port number.

channel: [output] The CAN port number. (1:CAN1, 2:CAN2, 3:Both)

mod: [output] ID mode (0:11 bit, 1:29 bit)

rtr: [output] Remote transmit request (0:disable, 1:enable)

dlc: [output] Data length (Range 0 ~ 8)

id[ID_LEN]: [output] The CAN ID (ID_LEN = 4)

data[DATA_LEN]: [output] Data (DATA_LEN = 8)

Return Status Code:

Value	**Return Value**
0	No data
>0	Get data

3.2.8. EMUCReceiveNonblock

Description: Receive multiple data from CAN to USB.

SYSTAX:

```
int EMUCReceiveNonblock (DATA_INFO *info, int cnt, unsigned int interval)
```

DATA_INFO struct:

```
typedef struct
{
        int com_port;
        int channel;
        int mod;
        int rtr;
        int dlc;
        unsigned char id[ID_LEN];
        unsigned char data[DATA_LEN];
} DATA_INFO;
```

Member:

com_port: [input] The virtual COM port number.

channel: [output] The CAN port number. (1:CAN1, 2:CAN2, 3:Both)

mod: [output] ID mode (0:11 bit, 1:29 bit)

rtr: [output] Remote transmit request (0:disable, 1:enable)

dlc: [output] Data length (Range 0 ~ 8)

id[ID_LEN]: [output] The CAN ID (ID_LEN = 4)

data[DATA_LEN]: [output] Data (DATA_LEN = 8)

cnt: [input]: Count of DATA_INFO structure

interval: [input] interval (ms) of receiving multiple data

Return Status Code:

Value	**Return Value**
0	No data
>0	Get data amount

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

+31-040-282-1818

Japan

jpsales@innodisk.com

+81-3-6667-0161

China

sales_cn@innodisk.com

+86-755-21673689

www.innodisk.com

December 16, 2016