

The logo for innodisk, featuring the word "innodisk" in white lowercase letters on a red rectangular background. A small red square is positioned to the right of the main rectangle.

**innodisk**

# **EMUC-B201**

USB to CANbus

User Manual

# Table of Contents

<b>1. Windows.....</b>	<b>1</b>
1.1. Install Driver .....	1
1.2. EMUC-B201 Test Utility.....	3
<b>2. Linux.....</b>	<b>4</b>
2.1. Install Driver .....	4
2.2. EMUC-B201 Test Utility.....	4
<b>3. Software API .....</b>	<b>5</b>
3.1. General Definition.....	5
3.2. CAN Channel and Baudrate Parameter.....	6
3.3. VER_INFO .....	6
3.4. DATA_INFO.....	6
3.5. Function Definition.....	7
3.6. Load Library (C/C++) .....	7
3.7. Function Description .....	8
3.7.1. EMUCShowVer .....	8
3.7.2. EMUCOpenDevice.....	9
3.7.3. EMUCCloseDevice .....	9
3.7.4. EMUCSetCAN.....	9
3.7.5. EMUCResetCAN.....	10
3.7.6. EMUCSend .....	10
3.7.7. EMUCReceive.....	11
<b>Contact us .....</b>	<b>13</b>

## Copyright Information

2005-2015 ©Innodisk Corporation. All Rights Reserved

Innodisk™ is trademark or registered trademark of Innodisk Corporation.

This document is subject to change and revision without notice. No part of this document may be reproduced in any form by any photographic, electronic, mechanical or other means, or used in any information storage and retrieval system, without prior written permission from Innodisk Corporation.

All other product and brand names in this document are trademarks or registered trademarks of their respective owners.

## 版權說明

2005-2015 ©宜鼎國際股份有限公司

Innodisk™ 是宜鼎國際股份有限公司之註冊商標。

本文件得不經通知即更改或修訂。本文件中出現任何文字敘述、文件格式、圖形、照片、方法及過程等內容，除另特別註明，版權均屬宜鼎國際股份有限公司所有，受到相關之智慧財產權保護法之保障。任何個人、法人或機構未經宜鼎國際股份有限公司的書面（包括電子文件）授權，不得以任何形式複製或引用本文件之全部或片段。

其他出現在本文件的品牌或產品乃歸屬原公司所有之商標或註冊。

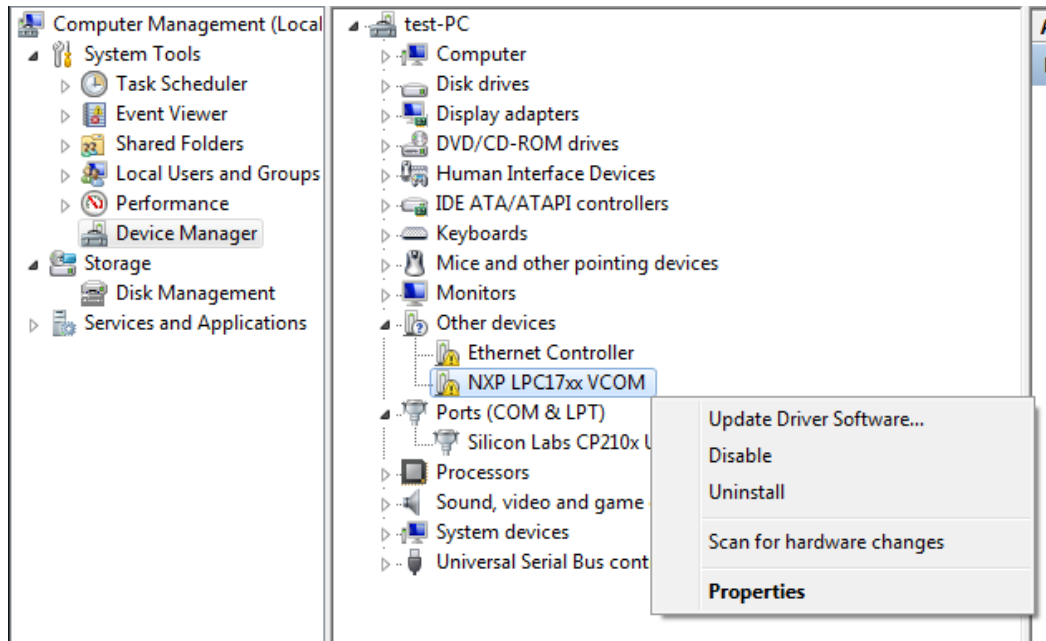
---

## 1. Windows

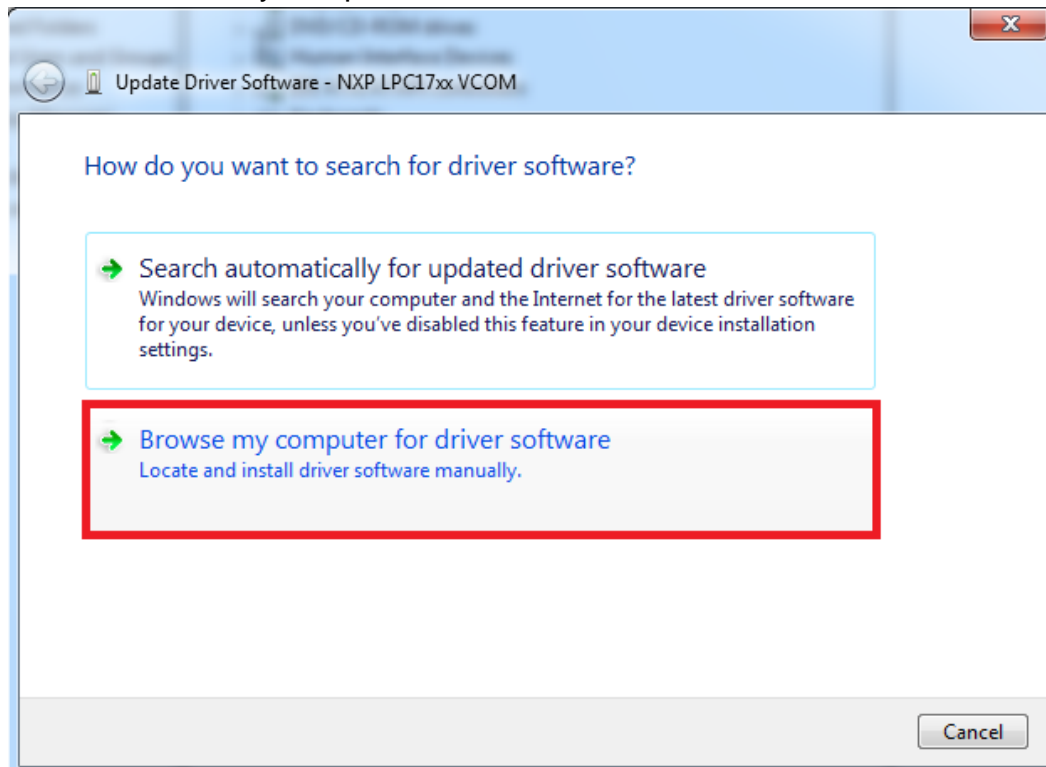
### 1.1. Install Driver

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”.

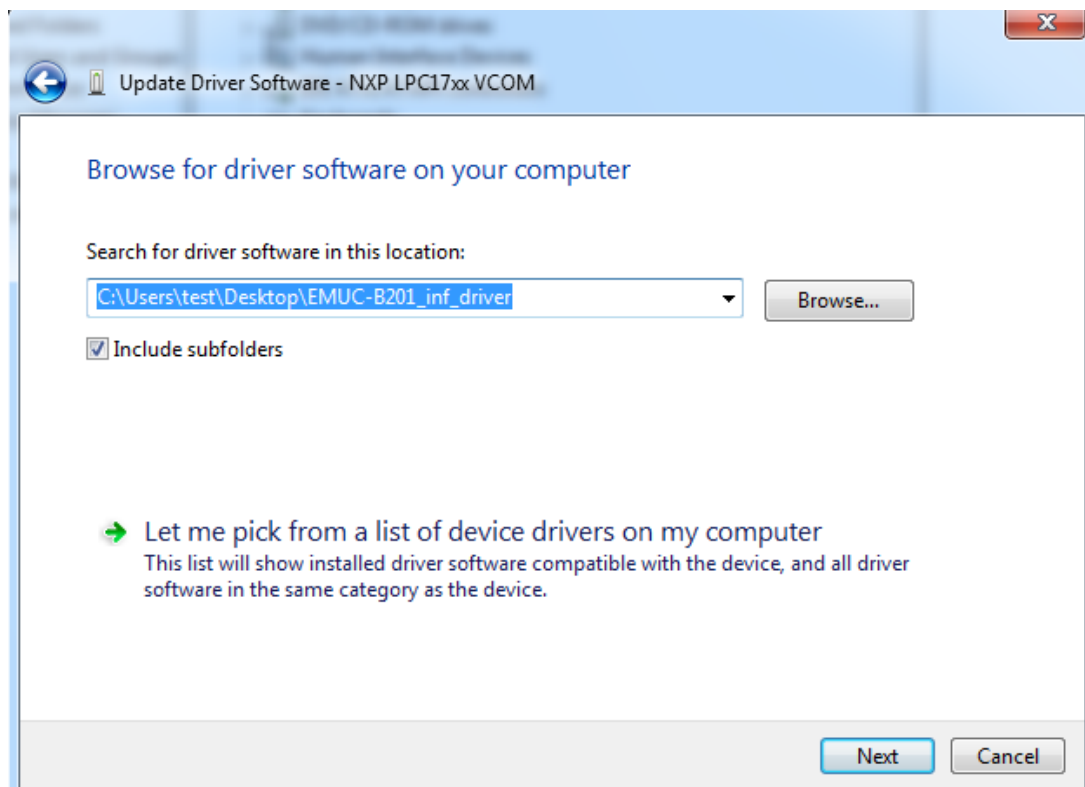
Click “Update Driver Software” to install driver.



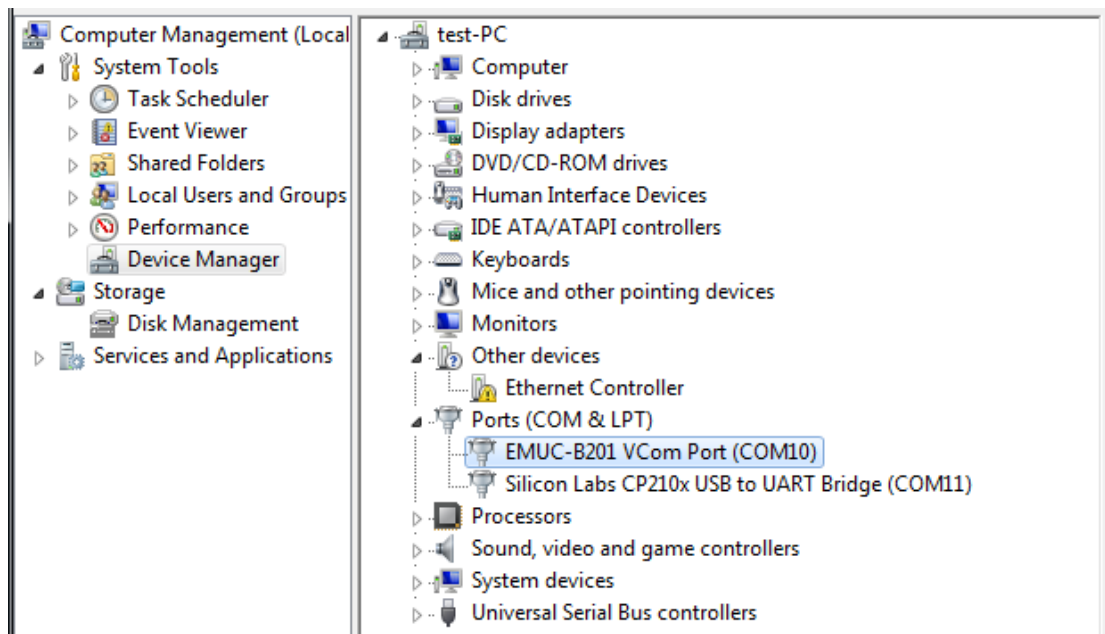
Select “Browse my computer for driver software”



Browse the path linking to EMUC-B201 driver.



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



## 1.2. EMUC-B201 Test Utility

You can use this tool to test EMUC-B201.

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 below messages.

```
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 ISDN 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="apparmor_parser"
[ 1963.682542] type=1400 audit(1446820607.258:67): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/usr/sbin/cupsd" pid=2916 comm="apparmor_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 SEMICONDUCTOR
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 not 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** as 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 interval 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    : AABBC
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]

```

### 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.

**NOTE:** Linux API supports the following port names.

"/dev/ttyS0", "/dev/ttyS1", "/dev/ttyS2", "/dev/ttyS3", "/dev/ttyS4", "/dev/ttyS5",  
"/dev/ttyS6", "/dev/ttyS7", "/dev/ttyS8", "/dev/ttyS9", "/dev/ttyS10", "/dev/ttyS11",  
"/dev/ttyS12", "/dev/ttyS13", "/dev/ttyS14", "/dev/ttyS15", "/dev/ttyUSB0",  
"/dev/ttyUSB1", "/dev/ttyUSB2", "/dev/ttyUSB3", "/dev/ttyUSB4", "/dev/ttyUSB5",  
"/dev/ttyAMA0", "/dev/ttyAMA1", "/dev/ttyACM0", "/dev/ttyACM1",  
"/dev/rfcomm0", "/dev/rfcomm1", "/dev/ircomm0", "/dev/ircomm1",  
"/dev/cuau0", "/dev/cuau1", "/dev/cuau2", "/dev/cuau3",  
"/dev/cuaU0", "/dev/cuaU1", "/dev/cuaU2", "/dev/cuaU3".

#### 3.1. General Definition

```

#define MAX_COM_PORT    16
#define MAX_BUF         512

```



```
#define ID_LEN      4
#define DATA_LEN   8
#define VER_LEN     16
```

### 3.2. CAN Channel and Baudrate Parameter

```
enum
{
    /* channel */
    EMUC_CH_1      = 1,
    EMUC_CH_2,
    EMUC_CH_BOTH,

    /* baudrate */
    EMUC_BAUDRATE_50K  = 3,
    EMUC_BAUDRATE_125K,
    EMUC_BAUDRATE_250K,
    EMUC_BAUDRATE_500K,
    EMUC_BAUDRATE_1M,
};
```

### 3.3. VER\_INFO

**Description:** Data structure is used for the API function “*EMUCShowVer*”.

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

### 3.4. DATA\_INFO

**Description:** Data structure is used for the API function “*EMUCReceive*” and “*EMUCSend*”.

```
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;
```

### 3.5. Function Definition

```
typedef int  (*EMUC_OPEN_DEVICE) (int port);  
typedef void (*EMUC_CLOSE_DEVICE) (int port);  
typedef int  (*EMUC_SHOW_VER)    (int port, VER_INFO *ver);  
typedef int  (*EMUC_SET_CAN)     (int port, int ch, int bdrate);  
typedef int  (*EMUC_RESET)      (int port);  
typedef int  (*EMUC_RECEIVE)    (DATA_INFO *info);  
typedef int  (*EMUC_SEND)       (DATA_INFO *info);
```

### 3.6. Load Library (C/C++)

```
static HMODULE      hDLL;  
static EMUC_OPEN_DEVICE  EMUCOpenDevice;  
static EMUC_CLOSE_DEVICE EMUCCloseDevice;  
static EMUC_SHOW_VER    EMUCShowVer;  
static EMUC_SET_CAN     EMUCSetCAN;  
static EMUC_RESET       EMUCReset;  
static EMUC_RECEIVE     EMUCReceive;  
static EMUC_SEND        EMUCSend;  
  
static int load_dll(void)  
{  
    hDLL = LoadLibrary("lib_emuc.dll");  
  
    if(!hDLL)  
        return FALSE;  
  
    EMUCOpenDevice  = (EMUC_OPEN_DEVICE)  GetProcAddress(hDLL,  
"EMUCOpenDevice");  
    EMUCCloseDevice = (EMUC_CLOSE_DEVICE) GetProcAddress(hDLL,  
"EMUCCloseDevice");
```

```

    EMUCShowVer    = (EMUC_SHOW_VER)    GetProcAddress(hDLL,
"EMUCShowVer");
    EMUCSetCAN     = (EMUC_SET_CAN)     GetProcAddress(hDLL,
"EMUCSetCAN");
    EMUCReset      = (EMUC_RESET)      GetProcAddress(hDLL,
"EMUCReset");
    EMUCReceive    = (EMUC_RECEIVE)    GetProcAddress(hDLL,
"EMUCReceive");
    EMUCSend       = (EMUC_SEND)       GetProcAddress(hDLL,
"EMUCSend");

    return TRUE;
}

```

## 3.7. Function Description

This chapter describes API functions and parameters.

### 3.7.1. EMUCShowVer

**Description:** Get firmware and library version.

#### SYNTAX:

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

#### Parameter:

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	EXCEPTION

### 3.7.2. EMUCOpenDevice

**Description:** Open virtual COM port.

**SYNTAX:**

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

**Parameter:**

port: [input] The virtual COM port number. (1:COM1, 2:COM2, 3:COM3...)

**Return Status Code:**

Value	Return Value
0	SUCCESS
1	COM_NOT_EXIST
5	COM_IN_USE

### 3.7.3. EMUCCloseDevice

**Description:** Close virtual COM port.

**SYNTAX:**

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

**Parameter:**

port: [input] The virtual COM port number. (1:COM1, 2:COM2, 3:COM3...)

### 3.7.4. EMUCSetCAN

**Description:** Set baud rate of CAN port.

**SYNTAX:**

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

**Parameter:**

port: [input] The virtual COM port number. (1:COM1, 2:COM2, 3:COM3...)

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

### 3.7.5. EMUCResetCAN

**Description:** Reset CAN port and clearing register without changing baudrate

**SYNTAX:**

```
int EMUCResetCAN(int por)
```

**Parameter:**

port: [input] The virtual COM port number. (1:COM1, 2:COM2, 3:COM3...)

**Return Status Code:**

Value	Return Value
0	SUCCESS
-1	RESET FAIL

### 3.7.6. EMUCSend

**Description:** Send data from USB to CAN.

**SYNTAX:**

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

**Parameter:**

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. (1:COM1, 2:COM2, 3:COM3...)

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)

**3.7.7. EMUCReceive**

**Description:** Receive data from CAN to USB.

**SYNTAX:**

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

**Parameter:**

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. (1:COM1, 2:COM2, 3:COM3...)

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

## 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](mailto:sales@innodisk.com)

### Branch Offices:

#### USA

[usasales@innodisk.com](mailto:usasales@innodisk.com)

+1-510-770-9421

#### Europe

[eusales@innodisk.com](mailto:eusales@innodisk.com)

+31-040-282-1818

#### Japan

[jpsales@innodisk.com](mailto:jpsales@innodisk.com)

+81-45-594-7581

#### China

[sales\\_cn@innodisk.com](mailto:sales_cn@innodisk.com)

+86-755-21673689

**[www.innodisk.com](http://www.innodisk.com)**

© 2015 Innodisk Corporation.

All right reserved. Specifications are subject to change without prior notice.

November 6, 2015