innodisk

EMUI-0D01

USB to 32bit DIO

User Manual

Rev 1.0

Revision History

Revision	Date	Description
1.0	2016/10/04	Initial Release

Table of Contents

1. Introduction

Innodisk EMUI-0D01 USB DIO card provides 32bit DIO grouped into four ports: A, B, C and D. It can connect with either mPCIe slot or USB pin header.
Each port can be configured by software to function as either input or output.
Bit4-Bit7 of Port D can be further configured as external interrupts. EMUI-0D01 saves port configurations (direction/status/interrupt) into EEPROM automatically and also can export or import configuration by software.

EMUI VCCIO logic level can be globally configured as 5V or 3.3V via DIP switch . Each DIO bit is buffered and capable of providing up to 24mA source/sink for 3.3V and 24mA source/sink for 5V.

Features:

- 32bit digital I/O in 4 ports (each port 8bit)
- Programmable I/O
- Selectable VCCIO 3.3V or 5V by DIP switch
- Buffered I/O
  - (Output 5V, 32mA source, 32mA sink)
  - (Output 3.3V, 24mA source, 24mA sink)
- 4 external interrupt with rising/falling edge on port D
- Keeps configuration after hardware reboot
- Supports 3rd mounting hole and USB Pin header for out-of-minicard installation
- Complies with EN61000-4-2 (ESD) Air-15kV, Contact-8kV
- Industrial temperature(-40 ºC to 85 ºC) operation
- 30µ ” golden finger, 3 years warranty
- Industrial design, manufactured in Innodisk Taiwan

Factory default setting

VCCIO Logic Level	5V It can be set to 3.3V by hardware DIP switch
Port Direction	Input
Port Status	Pull-low with 10kΩ resistor.
External interrupt	Disable
Interrupt Edge	Rising

2. Connector Pin Define

DB37 Male Connector Pin Define

Signal	Pin	Pin	Signal
GND	20	1	GND
PA0	21	2	GND
PA1	22	3	PD0
PA2	23	4	PD1
PA3	24	5	PD2
PA4	25	6	PD3
PA5	26	7	PD4
PA6	27	8	PD5
PA7	28	9	PD6
PB0	29	10	PD7
PB1	30	11	PC0
PB2	31	12	PC1
PB3	32	13	PC2
PB4	33	14	PC3
PB5	34	15	PC4
PB6	35	16	PC5
PB7	36	17	PC6
GND	37	18	PC7
		19	GND

40DP-1.25 Connector Pin Define

Signal	Pin	Pin	Signal
GND	1	2	GND
PC7	3	4	PB7
PC6	5	6	PB6
PC5	7	8	PB5
PC4	9	10	PB4
PC3	11	12	PB3
PC2	13	14	PB2
PC1	15	16	PB1
PC0	17	18	PB0
PD7/INT3	19	20	PB7

PD6/INT2	21	22	PA6
PD5/INT1	23	24	PA5
PD4/INT0	25	26	PA4
PD3	27	28	PA3
PD2	29	30	PA2
PD1	31	32	PA1
PD0	33	34	PA0
GND	35	36	GND
GND	37	38	GND
+3.3V(Reserve)	39	40	+3.3V(Reserve)

3.Control Bitmap

DIO Port Status Control

Bit 7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Px7	Px6	Px5	Px4	Px3	Px2	Px1	Px0

x=A, B, C, D

Bit[7:0] 0=Pull-down (Low), 1=Pull-up (High)

Example:

Port A[7:0] status 0xFF=1111 1111, 0x55=0101 0101, 0xAA=1010 1010

External Interrupt Control

Bit 7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INT3 Status	INT2 Status	INT1 Status	INT0 Status	INT3 Edge	INT2 Edge	INT1 Edge	INT0 Edge

Bit[7:4] 0=Disable, 1=Enable

Bit[0:3] 0=Falling Edge, 1=Rising Edge

Example:

0x11=0001 0001 means that INT0 is enabled and set as raising edge.

0x10=0001 0000 means that INT0 is enabled and set as falling edge.

0x66=0110 0110 means that INT1 and INT2 are enabled and both set as raising edge.

0x60=0110 0000 means that INT1 and INT2 are both enabled and set as falling edge

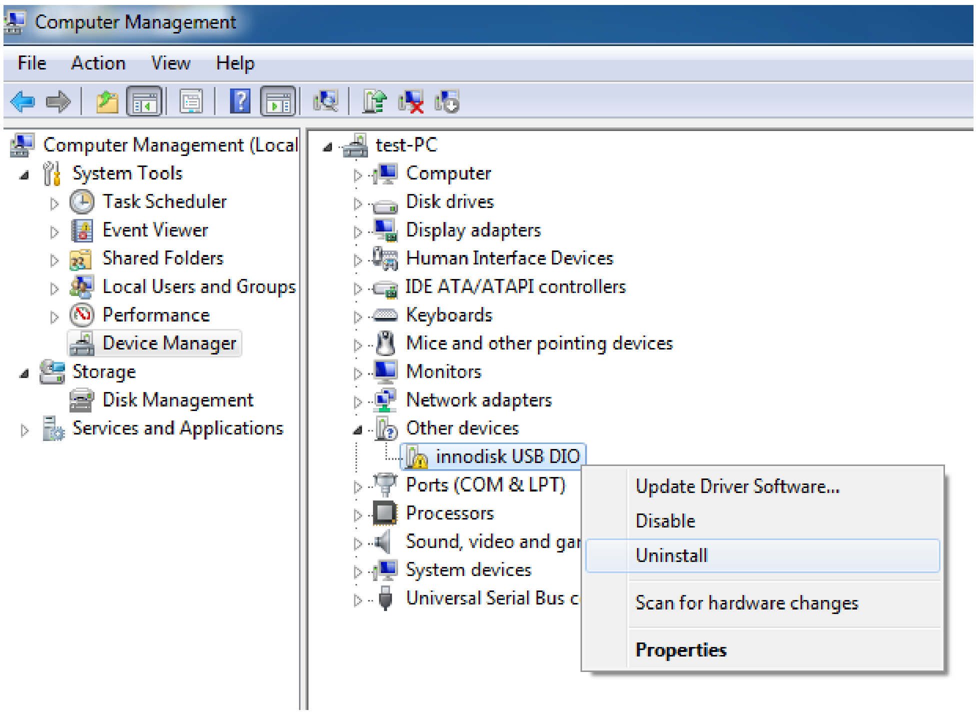
4. Windows OS

4.1. Installation

Step1.

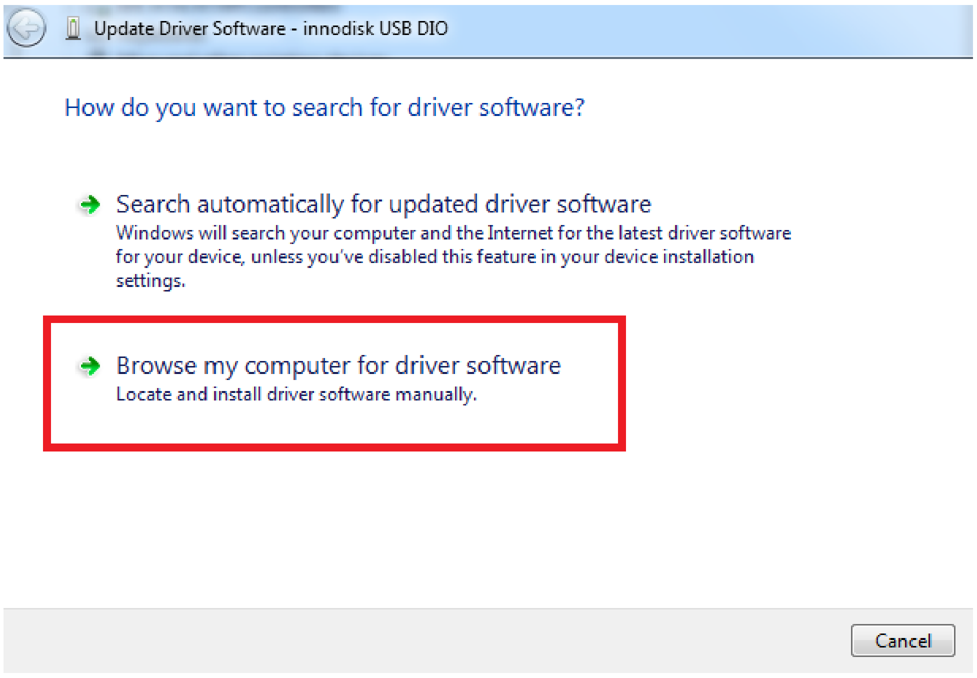
Install EMUI-0D01 either with mPCIe slot or USB pin header. The device named "innodisk USB DIO" can be found in "Device Manager".

Click "Update Driver Software" to install driver.



Step2.

Select "Browse my computer for driver software"


Update Driver Software - innodisk USB DIO

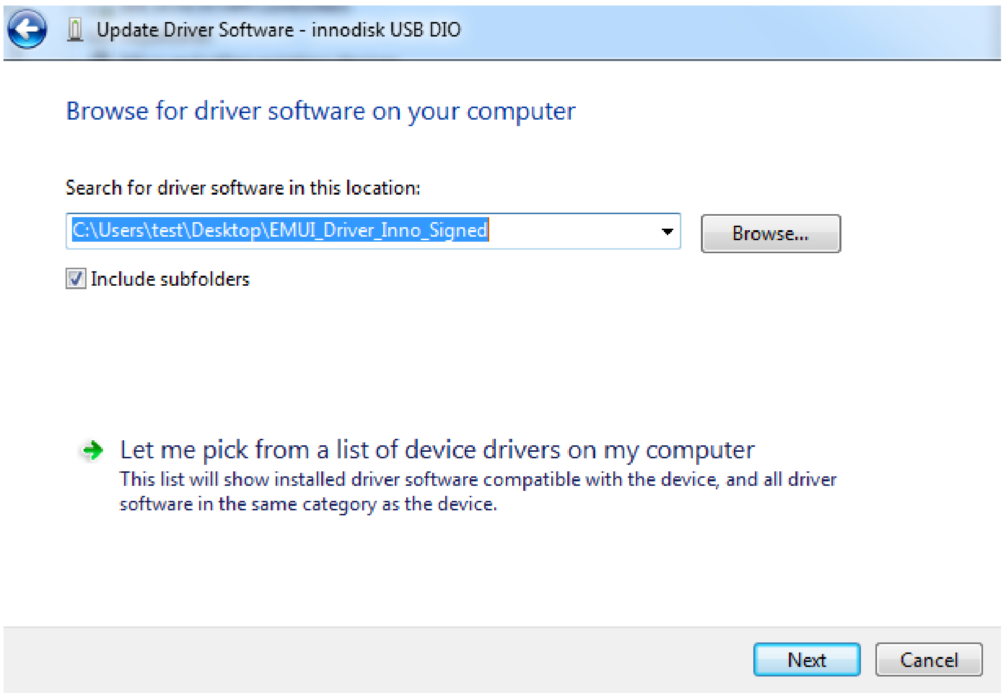
How do you want to search for driver software?

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.

Step3.

Browse the path linking to EMUI-0D01 driver.



Update Driver Software - innodisk USB DIO

Browse for driver software on your computer

Search for driver software in this location:

C:\Users\test\Desktop\EMUI_Driver_Inno_Signed

Browse...

☑ Include subfolders

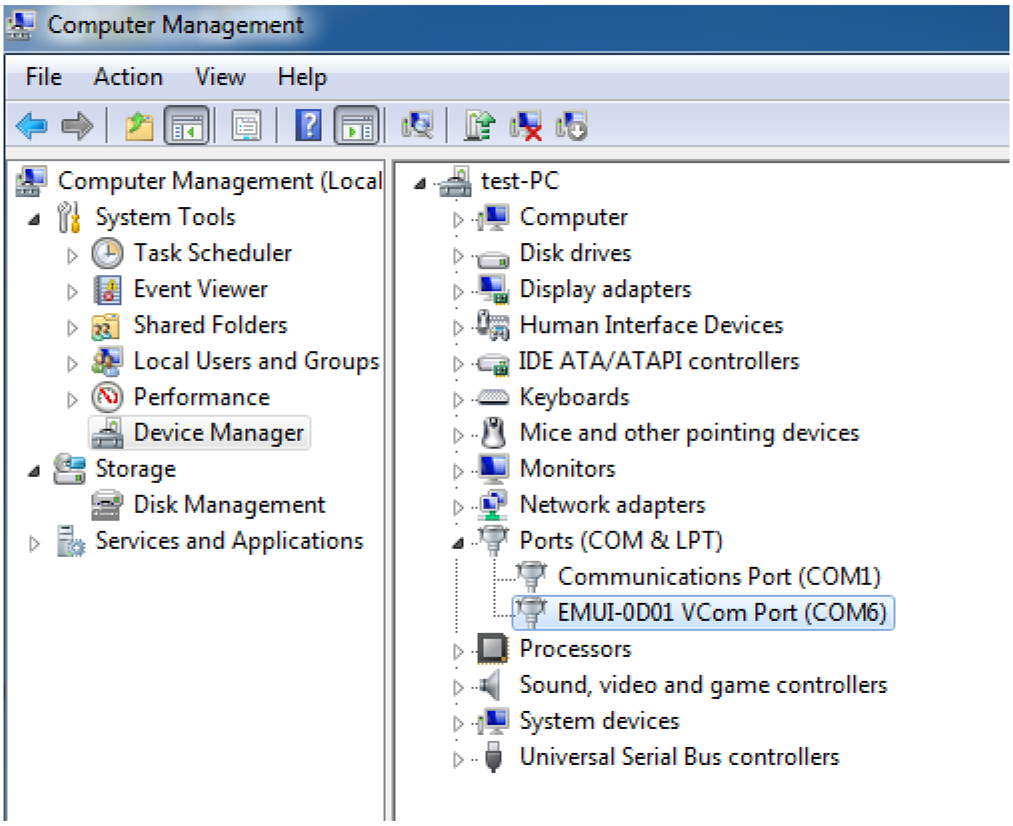
Let me pick from a list of device drivers on my computer

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

Next Cancel

Step4.

After installing driver, device can be recognized as a COM port by Windows CDC-ACM inbox driver (Compaq USB Modem Driver).



Computer Management

File Action View Help

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

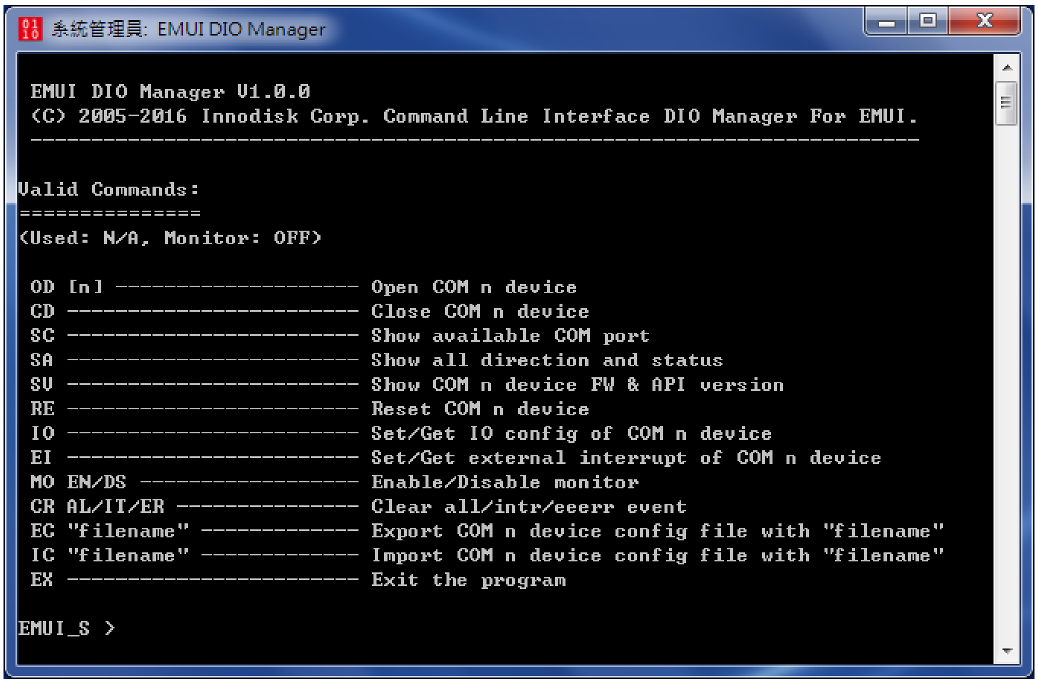
test-PC
- Computer
- Disk drives
- Display adapters
- Human Interface Devices
- IDE ATA/ATAPI controllers
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- Ports (COM & LPT)
  - Communications Port (COM1)
  - EMUI-0D01 VCom Port (COM6)
- Processors
- Sound, video and game controllers
- System devices
- Universal Serial Bus controllers

4.2. EMUI-0D01 Console Utility for Windows

You can use this console utility to test EMUI-0D01 in Windows.

Please refer to Section 3 Control Bitmap used in this utility.

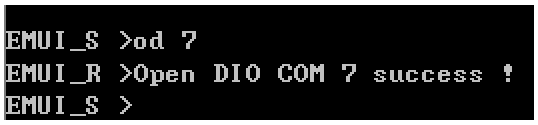
Every function in this utility we provide C code API library for programming.


```
系統管理員: EMUI DIO Manager

 EMUI DIO Manager V1.0.0
 (C) 2005-2016 Innodisk Corp. Command Line Interface DIO Manager For EMUI.
 --------------------------------------------------------------------------

Valid Commands:
===============
(Used: N/A, Monitor: OFF)

 OD [n] -------------------- Open COM n device
 CD ------------------------ Close COM n device
 SC ------------------------ Show available COM port
 SA ------------------------ Show all direction and status
 SV ------------------------ Show COM n device FW & API version
 RE ------------------------ Reset COM n device
 IO ------------------------ Set/Get IO config of COM n device
 EI ------------------------ Set/Get external interrupt of COM n device
 MO EN/DS ------------------ Enable/Disable monitor
 CR AL/IT/ER --------------- Clear all/intr/eeerr event
 EC "filename" ------------- Export COM n device config file with "filename"
 IC "filename" ------------- Import COM n device config file with "filename"
 EX ------------------------ Exit the program

EMUI_S >
```

4.2.1. **Command: OD [n]**

Description: Open virtual COM n port (n=real COM port number)

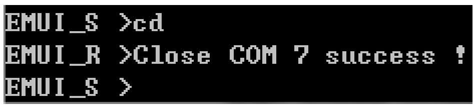
Example: Open COM 7


```
EMUI_S >od 7
EMUI_R >Open DIO COM 7 success !
EMUI_S >
```

4.2.2. **Command: CD**

Description: Close virtual COM n port (n=real COM port number)

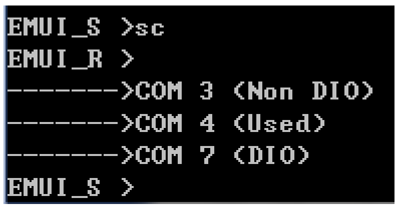
Example: Close an opened COM 7


```
EMUI_S >cd
EMUI_R >Close COM 7 success !
EMUI_S >
```

4.2.3. **Command: SC**

Description: Show all available COM ports in OS

Example:

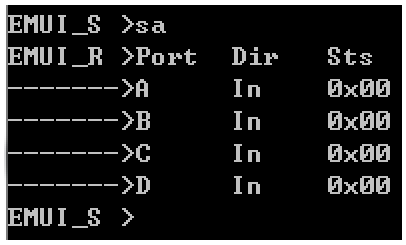


```
EMUI_S >sc
EMUI_R >
------->COM 3 (Non DIO)
------->COM 4 (Used)
------->COM 7 (DIO)
EMUI_S >
```

4.2.4. Command: SA

Description: Show directions and status of all 4 ports

Example:

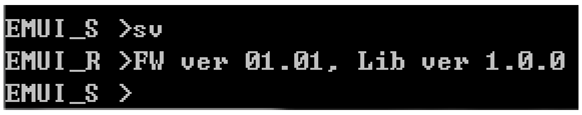


```
EMUI_S >sa
EMUI_R >Port  Dir   Sts
------->A     In    0x00
------->B     In    0x00
------->C     In    0x00
------->D     In    0x00
EMUI_S >
```

4.2.5. Command: SV

Description: Show FW version and API version

Example:

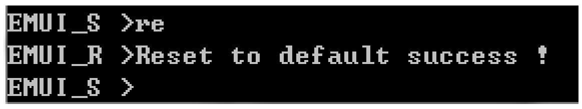


```
EMUI_S >sv
EMUI_R >FW ver 01.01, Lib ver 1.0.0
EMUI_S >
```

4.2.6. Command: RE

Description: Reset to factory default setting

Example:

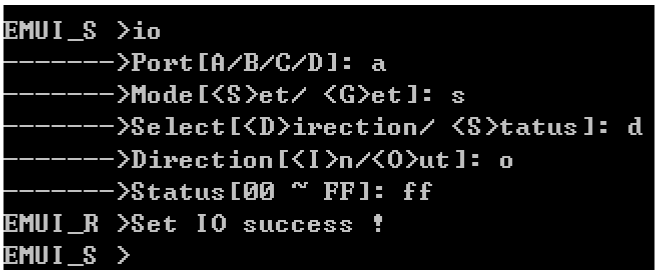


```
EMUI_S >re
EMUI_R >Reset to default success !
EMUI_S >
```

4.2.7. Command: IO

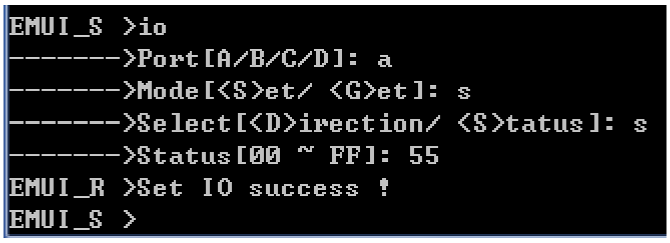
Description: Set or Get digital I/O direction and status

Example: Set Port A direction to output, 8bit status to 0xff (all pull-up)



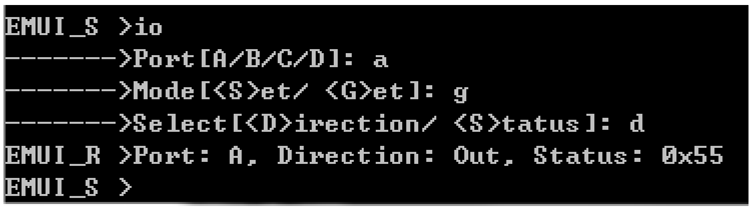
```
EMUI_S >io
------->Port[A/B/C/D]: a
------->Mode[<S>et/ <G>et]: s
------->Select[<D>irection/ <S>tatus]: d
------->Direction[<I>n/<O>ut]: o
------->Status[00 ~ FF]: ff
EMUI_R >Set IO success !
EMUI_S >
```

Example: Set Port A status to 0x55



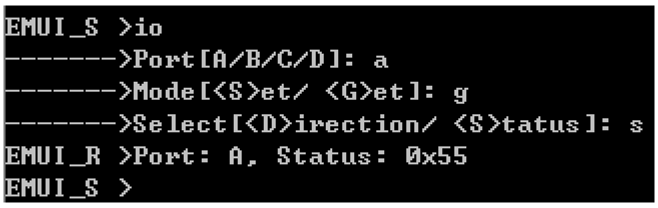
```
EMUI_S >io
------->Port[A/B/C/D]: a
------->Mode[<S>et/ <G>et]: s
------->Select[<D>irection/ <S>tatus]: s
------->Status[00 ~ FF]: 55
EMUI_R >Set IO success !
EMUI_S >
```

Example: Read Port A direction and status



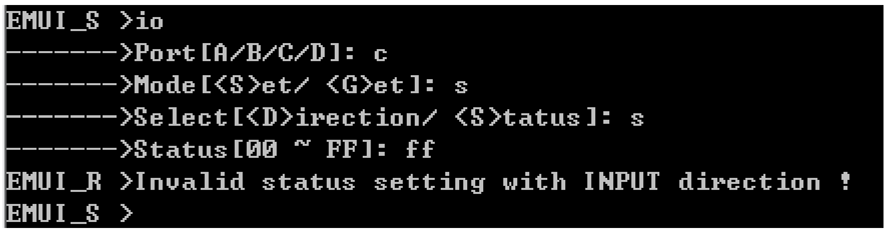
```
EMUI_S >io
------->Port[A/B/C/D]: a
------->Mode[<S>et/ <G>et]: g
------->Select[<D>irection/ <S>tatus]: d
EMUI_R >Port: A, Direction: Out, Status: 0x55
EMUI_S >
```

Example: Read Port A status



```
EMUI_S >io
------->Port[A/B/C/D]: a
------->Mode[<S>et/ <G>et]: g
------->Select[<D>irection/ <S>tatus]: s
EMUI_R >Port: A, Status: 0x55
EMUI_S >
```

Note: *Input port cannot set status. Its status must be given by external output port.*

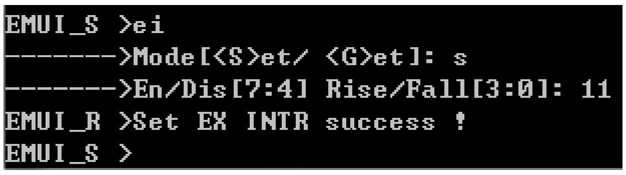


```
EMUI_S >io
------->Port[A/B/C/D]: c
------->Mode[<S>et/ <G>et]: s
------->Select[<D>irection/ <S>tatus]: s
------->Status[00 ~ FF]: ff
EMUI_R >Invalid status setting with INPUT direction !
EMUI_S >
```

4.2.8. **Command: EI**

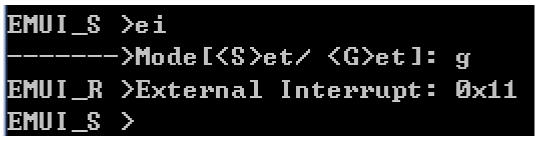
Description: Set or Get external interrupt configuration of Port D Bit4-Bit7

Example: Enable INT0 (Bit4 of Port D) and set it to rising edge.



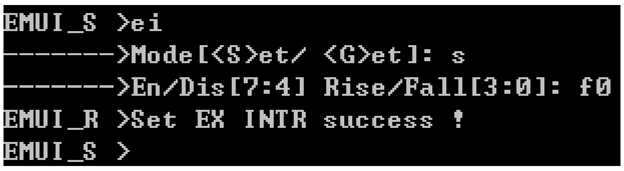
```
EMUI_S >ei
------->Mode[<S>et/ <G>et]: s
------->En/Dis[7:4] Rise/Fall[3:0]: 11
EMUI_R >Set EX INTR success !
EMUI_S >
```

Example: Get external interrupt configuration



```
EMUI_S >ei
------->Mode[<S>et/ <G>et]: g
EMUI_R >External Interrupt: 0x11
EMUI_S >
```

Example: Enable all external interrupt and set it to falling edge.



```
EMUI_S >ei
------->Mode[<S>et/ <G>et]: s
------->En/Dis[7:4] Rise/Fall[3:0]: f0
EMUI_R >Set EX INTR success !
EMUI_S >
```

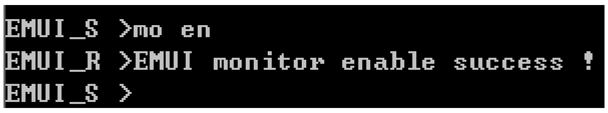
Note: *When enabling external interrupt, Port D will be set to input automatically, and bit0-bit3 only can be input as well.*

4.2.9. **Command: MO EN/DS**

Description: Enable/Disable monitoring of below events

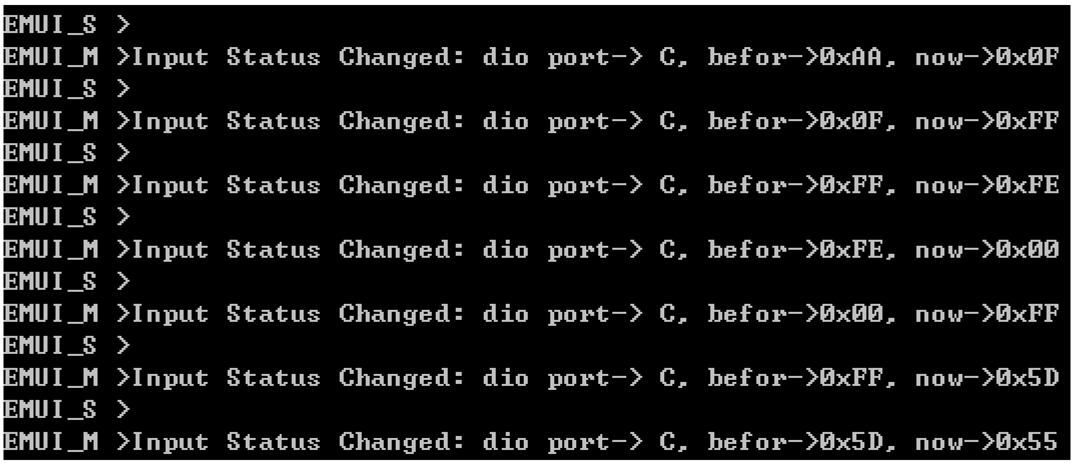
1. **Input status change:** Send eventonce immediately.
2. **Interrupt:** Send event immediately when triggered, then continue sending event every 2 sec.
3. **EEPROM Error:** Send event every 5 sec after the module power-on. If you clear this event, it will not send event anymore until next power-on.

Example:



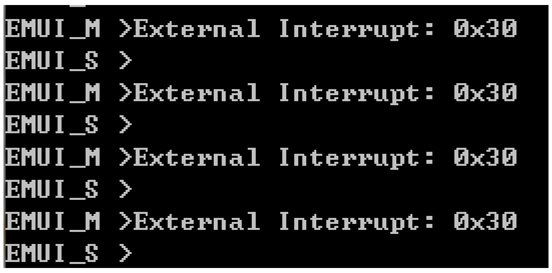
```
EMUI_S >mo en
EMUI_R >EMUI monitor enable success !
EMUI_S >
```

Example: Status change occurs

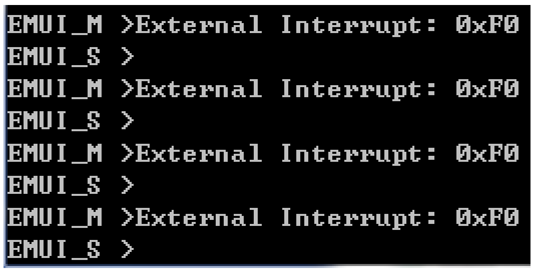


```
EMUI_S >
EMUI_M >Input Status Changed: dio port-> C, befor->0xAA, now->0x0F
EMUI_S >
EMUI_M >Input Status Changed: dio port-> C, befor->0x0F, now->0xFF
EMUI_S >
EMUI_M >Input Status Changed: dio port-> C, befor->0xFF, now->0xFE
EMUI_S >
EMUI_M >Input Status Changed: dio port-> C, befor->0xFE, now->0x00
EMUI_S >
EMUI_M >Input Status Changed: dio port-> C, befor->0x00, now->0xFF
EMUI_S >
EMUI_M >Input Status Changed: dio port-> C, befor->0xFF, now->0x5D
EMUI_S >
EMUI_M >Input Status Changed: dio port-> C, befor->0x5D, now->0x55
```

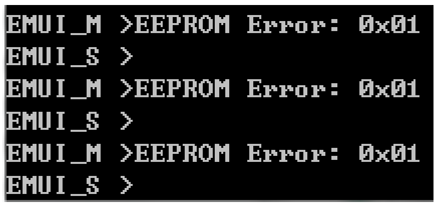
Example: Interrupt occurs on INT0 and INT1


```
EMUI_M >External Interrupt: 0x30
EMUI_S >
EMUI_M >External Interrupt: 0x30
EMUI_S >
EMUI_M >External Interrupt: 0x30
EMUI_S >
EMUI_M >External Interrupt: 0x30
EMUI_S >
```

Example: Interrupt occurs on all INT port (INT0-INT3)


```
EMUI_M >External Interrupt: 0xF0
EMUI_S >
EMUI_M >External Interrupt: 0xF0
EMUI_S >
EMUI_M >External Interrupt: 0xF0
EMUI_S >
EMUI_M >External Interrupt: 0xF0
EMUI_S >
```

Example: EEPROM error occurs


```
EMUI_M >EEPROM Error: 0x01
EMUI_S >
EMUI_M >EEPROM Error: 0x01
EMUI_S >
EMUI_M >EEPROM Error: 0x01
EMUI_S >
```

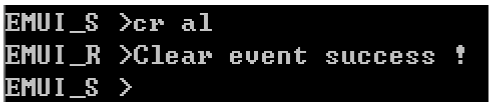
4.2.10. **Command: CR AL/IT/ER**

Description: Clear interrupt or EEPROM error events

You can clear persistent interrupt events or EEPROM error events.

AL=all event, IT=interrupt event, ER=EEPROM error

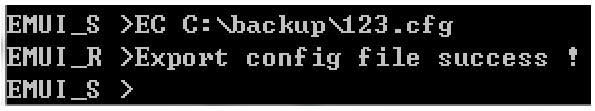
Example: Clear all events


```
EMUI_S >cr al
EMUI_R >Clear event success !
EMUI_S >
```

4.2.11. **Command: EC**

Description: Export configuration

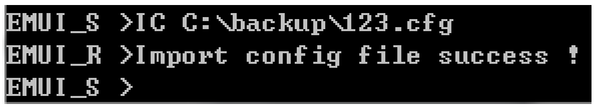
Example:


```
EMUI_S >EC C:\backup\123.cfg
EMUI_R >Export config file success !
EMUI_S >
```

4.2.12. **Command: IC**

Description: Import configuration

Example:



```
EMUI_S >IC C:\backup\123.cfg
EMUI_R >Import config file success !
EMUI_S >
```

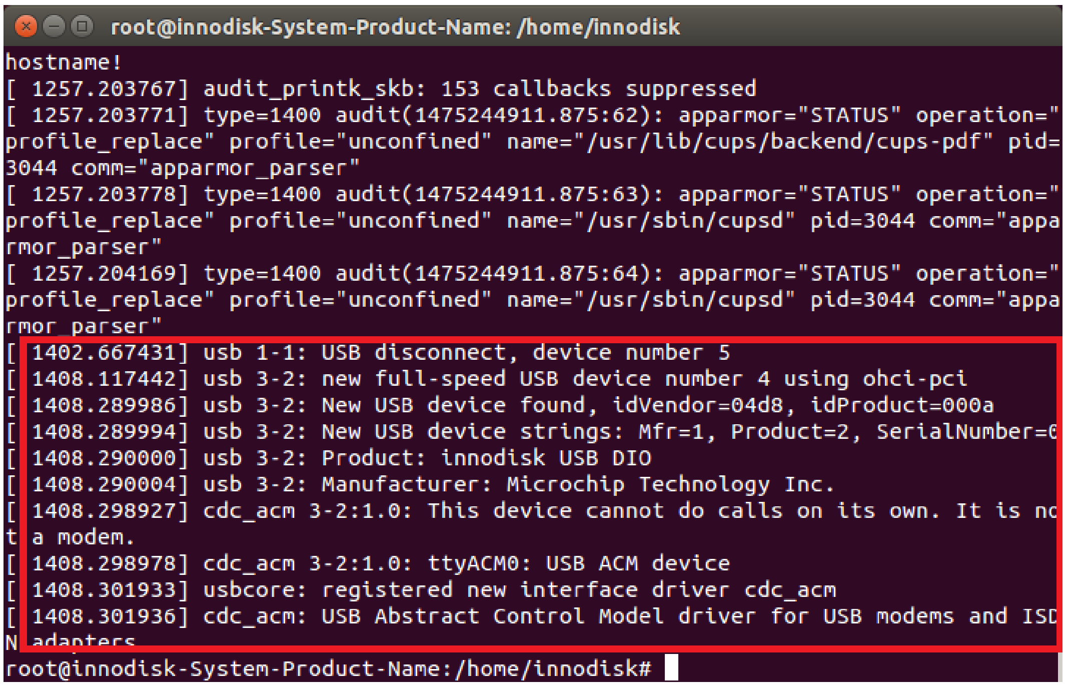
5. Linux OS

5.1. Installation

Install EMUI-0D01 either with mPCIe slot or 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.



```
hostname!
[ 1257.203767] audit_printk_skb: 153 callbacks suppressed
[ 1257.203771] type=1400 audit(1475244911.875:62): apparmor="STATUS" operation="
profile_replace" profile="unconfined" name="/usr/lib/cups/backend/cups-pdf" pid=
3044 comm="apparmor_parser"
[ 1257.203778] type=1400 audit(1475244911.875:63): apparmor="STATUS" operation="
profile_replace" profile="unconfined" name="/usr/sbin/cupsd" pid=3044 comm="appa
rmor_parser"
[ 1257.204169] type=1400 audit(1475244911.875:64): apparmor="STATUS" operation="
profile_replace" profile="unconfined" name="/usr/sbin/cupsd" pid=3044 comm="appa
rmor_parser"
[ 1402.667431] usb 1-1: USB disconnect, device number 5
[ 1408.117442] usb 3-2: new full-speed USB device number 4 using ohci-pci
[ 1408.289986] usb 3-2: New USB device found, idVendor=04d8, idProduct=000a
[ 1408.289994] usb 3-2: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[ 1408.290000] usb 3-2: Product: innodisk USB DIO
[ 1408.290004] usb 3-2: Manufacturer: Microchip Technology Inc.
[ 1408.298927] cdc_acm 3-2:1.0: This device cannot do calls on its own. It is no
t a modem.
[ 1408.298978] cdc_acm 3-2:1.0: ttyACM0: USB ACM device
[ 1408.301933] usbcore: registered new interface driver cdc_acm
[ 1408.301936] cdc_acm: USB Abstract Control Model driver for USB modems and ISD
N adapters
root@innodisk-System-Product-Name:/home/innodisk#
```

5.2. EMUI-0D01 Console Utility for Linux

You can use this console utility to test EMUI-0D01 in Linux.

Please refer to Section 3 Control Bitmap used in this utility.

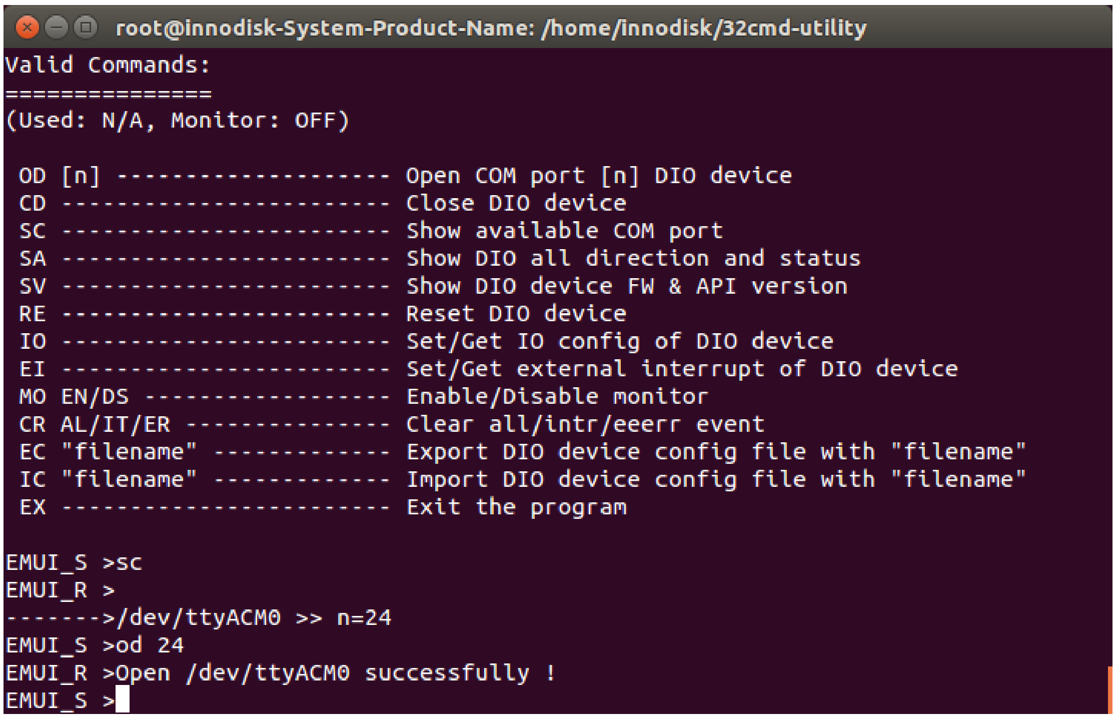
Every function in this utility we provide C code API library for programming.

Linux console utility is almost the same as Windows version.

The only difference is that serial port name is ttyACM instead of COM.

You can use "SC" command to check the index of available serial ports and open it to use EMUI-0D01.

Example: The index of ttyACM0 is 24 in EMUI API.



```
root@innodisk-System-Product-Name: /home/innodisk/32cmd-utility
Valid Commands:
===============
(Used: N/A, Monitor: OFF)

 OD [n] -------------------- Open COM port [n] DIO device
 CD ------------------------ Close DIO device
 SC ------------------------ Show available COM port
 SA ------------------------ Show DIO all direction and status
 SV ------------------------ Show DIO device FW & API version
 RE ------------------------ Reset DIO device
 IO ------------------------ Set/Get IO config of DIO device
 EI ------------------------ Set/Get external interrupt of DIO device
 MO EN/DS ------------------ Enable/Disable monitor
 CR AL/IT/ER --------------- Clear all/intr/eeerr event
 EC "filename" ------------- Export DIO device config file with "filename"
 IC "filename" ------------- Import DIO device config file with "filename"
 EX ------------------------ Exit the program

EMUI_S >sc
EMUI_R >
------->/dev/ttyACM0 >> n=24
EMUI_S >od 24
EMUI_R >Open /dev/ttyACM0 successfully !
EMUI_S >
```

6. Software API

EMUI API is based on a dynamic library (DLL) in Windows and static library (.a) in Linux to control EMUI-0D01.

lib_emui.h includes declaration and data structure requested for programming.

Windows API must come with libwinpthread-1.dll to use for the monitor function.

6.1. COM Port Selection

EMUI-0D01 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

EMUI-0D01 supports the following COM names. The names map “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	rfcomm0
27	rfcomm1	28	ircomm0	29	Ircomm1
30	cuau0	31	Cuau1	32	Cuau2
33	cuau3	34	cuaU0	35	cuaU1
36	cuaU2	37	cuaU3		

6.2. Function Description

This chapter describes API functions and parameters.

6.2.1. EMUIShowVer

Description: Get firmware and library version.

SYSTAX:

```
int EMUIShowVer(VER_INFO *ver_info)
```

VER_INFO Struct:

```
typedef struct
{
    int com_port;

    char fw [VER_LEN];
    char api[VER_LEN];
} VER_INFO;
```

Member:

com_port: [input] The virtual COM port number

fw: [output] Firmware version

api: [output] API version

Return Code:

Value	Description
0	Success
1	Fail

6.2.2. **EMUIOpenDevice**

Description: Open virtual COM port.

SYSTAX:

```
Int EMUIOpenDevice(int com_port)
```

Member:

com_port: [input] The virtual COM port number

Return Code:

Value	Description
0	Success
1	Out of range --> com_port > 255
2 (Windows only)	COM port does not exist
5 (Windows only)	COM port was already used
6	Non EMUI DIO
7	EMUI DIO was already opened

6.2.3. **EMUICloseDevice**

Description: Close virtual COM port.

SYSTAX:

```
int EMUIOpenDevice(int com_port)
```

Member:

com_port: [input] The virtual COM port number

Return Code:

Value	Description
0	Success
1	COM port was not opened / Non EMUI DIO

6.2.4. EMUIIOConfig

Description: Set/Get digital I/O direction and status.

SYSTAX:

```
Int EMUIIOConfig (IO_INFO *io_info)
```

IO_INFO Struct:

```
typedef struct
{
    /* Set & Get config */
    int com_port;
    int dio_port;
    int mode;

    /* Set config */
    int dir_set;
    unsigned char sts_set;

    /* Get return info */
    GET_RTN    get_io_rtn;
} IO_INFO;
```

Member:

com_port: [input] The virtual COM port number

dio_port: [input] PortA=1, PortB=2, PortC=3, PortD=4

mode: [input] Set_Direction =1, Get_Direction=2, Set_Status=3, Get_Status=4

dir_set: [input] output=0, input=1

sts_set: [input] Low=0, High=1, don’t care when direction=1(input)

get_io_rtn: [output] use in mode 2 and mode 4 “get” functions, return dio_port, direction, status

```
typedef struct
{
    /* IO config */
    int    dio_port;
    int    direction;
    unsigned char status;

    /* External interrupt */
    unsigned char ex_intr;
} GET_RTN;
```

Return Code:

Value	Description
0	Success
1	Fail
2	Input port cannot set status

6.2.5. EMUIExIntrConfig

Description: Set/Get external interrupt configuration of Port D Bit4-Bit7.

SYSTAX:

```
Int EMUIExIntrConfig(EXINTR_INFO *exintr_info)
```

EXINTR Struct:

```
typedef struct
{
    /* Set & Get config */
    int com_port;
    int mode;

    /* Set config */
    unsigned char exintr_set;       /* bit [3:0]: set edge
                                     * bit [7:4]: set enable or disable
                                     */
    /* Get return info */
    GET_RTN    get_exintr_rtn;
} EXINTR_INFO;
```

Member:

com_port: [input] The virtual COM port number

mode: [input] Set_Ext_INTR=1, Get_Ext_INTR=2

exintr_set: [input] bit [7:4]: set enable or disable, bit [3:0]: set edge

bit[7:4]: Disable=0, Enable=1

bit[3:0]: Falling=0, Rising=1

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INT3	INT2	INT1	INT0	INT3_Edge	INT2_Edge	INT1Edge	INT0_Edge

get_exintr_rtn: [output] use in mode 2 "get" function, return ex_intr

```
typedef struct
{
    /* IO config */
    int    dio_port;
    int    direction;
    unsigned char status;

    /* External interrupt */
    unsigned char ex_intr;
} GET_RTN;
```

Return Code:

Value	Description
0	Success
1	Fail

6.2.6. EMUIReset

Description: Reset EMUI to factory default setting.

SYSTAX:

```
Int EMUIReset(int com_port)
```

Member:

com_port: [input] The virtual COM port number

Return Code:

Value	Description
0	Success
1	Fail

6.2.7. EMUIMonitor (call back function)

Description: Enable/Disable monitoring of events below.

1. **Input status change:** Send event once immediately
2. **Interrupt:** Send event immediately when triggered, then continue sending event every 2 sec

3. **EEPROM Error (used to store configuration):** Send event every 5 sec after the module power on. If you clear this event, it will not send event anymore until next power on.

SYSTAX:

```
Int EMUIMonitor(MONITOR_INFO *monitor_info)
```

Monitor_INFO struct:

```
typedef struct
{
    int mode;
    int com_port;

    bool is_sts[DIO_PORT_NUM];
    bool is_intr;
    bool is_eeerr;

    unsigned char rtn_sts_bfr[DIO_PORT_NUM];
    unsigned char rtn_sts_now[DIO_PORT_NUM];

    unsigned char rtn_intr;
    unsigned char rtn_eeerr;

    MONITOR_CB_FUNC    monitor_cb_func;
} MONITOR_INFO;
```

Member:

mode: [input] disable=0, enable=1
com_port: [input] The virtual COM port number
is_sts: [output] 0=false, 1=true, return rtn_sts_bfr and rtn_sts_now if true.
is_intr: [output] 0=false, 1=true, return rtn_intr if true
is_eeerr: [output] 0=false, 1=true, return rtn_eeerr if true
monitor_cb_func: [input] register a call back function as below:

```
typedef void (*MONITOR_CB_FUNC) (void *ptr);
```

Return Code:

Value	**Description**
0	Success

1	Fail
2	COM port was not opened / Non EMUI DIO

6.2.8. **EMUIClearEvent**

Description: Clear interrupt or EEPROM error events.

SYSTAX:

```
Int EMUIClearEvent(EVENT_INFO *event_info)
```

EVENT_INFO struct:

```
typedef struct
{
    int mode;
    int com_port;
} EVENT_INFO;
```

Member:

mode: [input] All_Event=0, INTR_Event =1, EE_Err_Event=2

com_port: [input] The virtual COM port number

Return Code:

Value	**Description**
0	Success
1	Fail

6.2.9. **EMUIExpCfg**

Description: Export configuration.

SYSTAX:

```
Int EMUIExpCfg (CFG_FILE_INFO *cfg_file_info)
```

CFG_FILE_INFO Struct:

```
typedef struct
{
    int      com_port;
    char    file_name[MAX_FILE_NAME_NUM];
} CFG_FILE_INFO;
```

Member:

com_port: [input] The virtual COM port number

file_name: [input] File name and path

Return Code:

Value	**Description**
0	Success
1	Fail

6.2.10. **EMUIImpCfg**

Description: Import configuration.

SYSTAX:

```
Int EMUIImpCfg(CFG_FILE_INFO *cfg_file_info)
```

CFG_FILE_INFO Struct:

```
typedef struct
{
    int      com_port;
    char     file_name[MAX_FILE_NAME_NUM];
} CFG_FILE_INFO;
```

Member:

com_port: [input] The virtual COM port number

file_name: [input] File name and path

Return Code:

Value	**Description**
0	Success
1	Fail

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

October 14, 2016