DMX-100 USER Manual

Mini PCIe Digital I/O Card, 32-bit GPIO/Isolated DIO
-40°C to 85°C Extended Operating Temperature

Vecow

1.3.0 Edition 20190725

Record of Revision

Version	Date	Page	Description	Remark
0.10	11/21/2018	All	Preliminary Release	
1.0	11/22/2018	All	Official Release	
1.1	01/21/2019	5	Update	
1.2	07/18/2019	12	Update	
1.3	07/25/2019	9	Update	

Disclaimer

This manual is released by Vecow Co., Ltd. for reference purpose only. All product offerings and specifications are subject to change without prior notice. It does not represent commitment of Vecow Co., Ltd. Vecow shall not be liable for direct, indirect, special, incidental, or consequential damages arising out of the use of the product or documentation or any infringements upon the rights of third parties, which may result from such use.

Declaration of Conformity

FCC

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy, and if it is not installed and used in accordance with the instruction manual, it may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

CE

The products described in this manual complies with all applicable European Union (CE) directives if it has a CE marking. For computer systems to remain CE compliant, only CE-compliant parts may be used. Maintaining CE compliance also requires proper cable and cabling techniques.

Copyright and Trademarks

Order Information

Part Number	Description
DMX-100	Mini PCIe 32-bit GPIO Card
DMX-110	Mini PCIe 32-bit GPIO Card with Isolated DIO Expansion Card

Table of Contents

1

GENERAL INTRODUCTION

1.1 Overview

Vecow DMX-100 Series is a Mini-PCIe Digital I/O module supporting -40°C to 85°C operating temperature. DMX-100 offers programmable GPIO; DMX-110 offers Isolated DIO control. DMX-100 provides 32-bit GPIO divided into four ports : A, B, C, D. Each port can be independently set as input, output, interrupt or counter function. Interrupt function saves processing time of repeatedly checking data and counter function allows status monitoring. Moreover, DMX-100 offers timer function for status reporting. DMX-110 provides 16-bit DI and 16-bit DO grouped into DIO1 and DIO2. DIO1 and DIO2 are isolated DIO expansion cards each with 8-bit DI and 8-bit DO. Each Isolated card can be chosen as Sink/Source Mode independently by SW/HW. Vecow DMX-100 Series Mini PCIe digital I/O card is your smart solution for any embedded applications.

1.2 Features

- 4KV Isolated Protection for each channel
- -40°C to 85°C Operating Temperature
- 32-bit Programmable signals
- 4 ports digital signal, each with 8-bit GPIO
- Onboard Interrupt supports Rising/Falling Edge
- Programmable Timer and Counter for status monitoring
- Isolated DIO Control : Sink Mode/Source Mode
- Supports Wide Range I/O Voltage

1.3 Product Specification

1.3.1 Specifications of DMX-100

DMX-100	
Interface	USB 2.0
Number of port	4 (8-bit GPIO/each)
Power Input	3.3V DC-in
I/O Voltage	3.3V
Software Support	
OS	Windows 10, Windows 7, Linux
Mechanical	
Dimension (W x L)	• DMX-100-M : 30.0mm x 50.9mm (1.18" x 2.00") • Cable Length : 250mm (9.84")
Weight	160 g
Environment	
Operating Temperature	-40°C to 85°C (-40°F to 185 °F)
Storage Temperature	-40°C to 85°C (-40°F to 185°F)
Humidity	5% to 95% Humidity, non-condensing
Relative Humidity	95% @ 85°C
EMC	CE, FCC

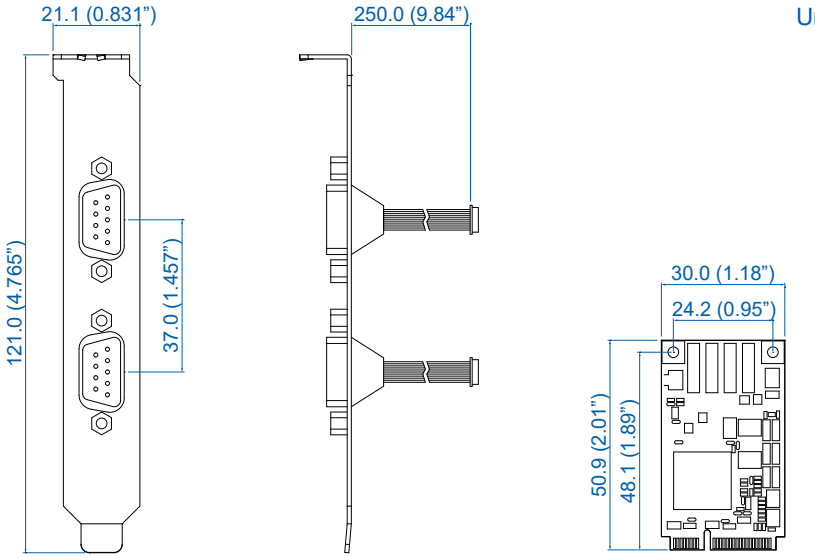
1.3.2 Specifications of DMX-110

DMX-110	
Interface	USB 2.0
Number of port	• Input : 4 (8-bit GPIO/each) • Output : 2 with 16-bit Isolated DIO
Power Input	3.3V DC-in
I/O Voltage	• Sink Mode : 6V to 40V • Source Mode : 6V to 48V
Software Support	
OS	Windows 10, Windows 7, Linux
Mechanical	
Dimension (W x L)	• DMX-100-M : 30.0mm x 50.9mm (1.18" x 2.00") • DMX-100-E : 45.0mm x 55.0mm (1.77" x 2.16") • Cable Length : 250mm (9.84")
Weight	250 g
Environment	
Operating Temperature	-40°C to 85°C (-40°F to 185 °F)
Storage Temperature	-40°C to 85°C (-40°F to 185°F)
Humidity	5% to 95% Humidity, non-condensing
Relative Humidity	95% @ 85°C
EMC	CE, FCC

1.4 Mechanical Dimension

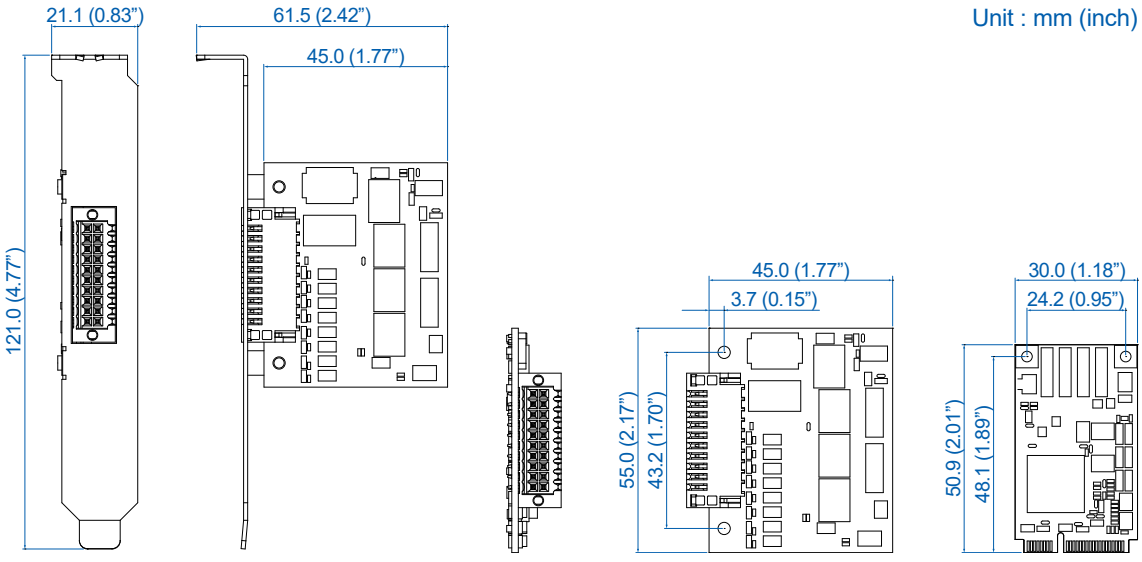
1.4.1 Dimensions of DMX-100

Unit : mm (inch)



1.4.2 Dimensions of DMX-110

Unit : mm (inch)



2

GETTING TO KNOW YOUR DMX-100

2.1 Packing List

2.1.1 DMX-100

Item	Description	Qty
1	DMX-100 GPIO module which contains DMX-100-M	1
2	DMX-100 Accessory box which contains PCIe Bracket with 2 GPIO Cables Screws	2 (Below)

Item	Description	Outlook	Usage	P/N	Qty
1	PHILLPIS M2.5x6L, Ni		Mini PCIe slot	53-2426906-30B	2

2.1.2 DMX-110

Item	Description	Qty
1	DMX-110 Isolated DIO module which contains DMX-100-M DMX-100-E with PCIe Bracket	1 2
2	DMX-110 Accessory box which contains Screws & Cable & Terminal block	(Below)

Item	Description	Outlook	Usage	P/N	Qty
1	PHILLPIS M2.5x6L, Ni		Mini PCIe slot	53-2426906-30B	2
2	Sink/Source Mode control cable		DMX-100-E to DMX-100-M	61-1300001-0DA	2
3	DIO cable		DMX-100-E to DMX-100-M	61-13U0001-0DA	4
4	Terminal block 20-pin (2.54mm)		Isolated DIO	51-2112R20-S1D	2

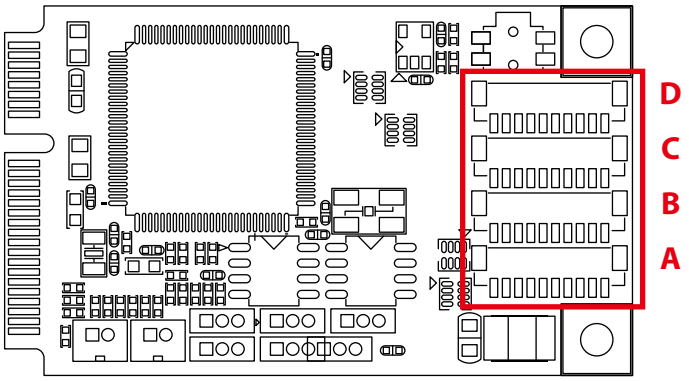
2.2 DMX-100-M I/O Function

In Vecow DMX-100-M, all I/O connectors are located on TOP side. Most of the general connections to computer interface, such as SMBus, JTAG, GPIO and control signals.

2.2.1 GPIO

The MCU offers 32 ports programmable I/O. (3.3V Level)

- 32bit digital I/O in 4 ports (Port A-D each port 8bit).
- If the GPIO is logic high, it indicates that the mapping MCU GPIO pin is logic high level.
- If the GPIO is logic low, it indicates that the mapping MCU GPIO pin is logic low level.
- Standard driving 5mA



GPIO Connectors definition :

Port A

Pin No.	Description
1	PTB16
2	PTB17
3	PTB18
4	PTB19
5	PTB20
6	PTB21
7	PTB22
8	PTB23
9	+VDIO (3.3v)
10	GND

Port B

Pin No.	Description
1	PTC0
2	PTC1
3	PTC2
4	PTC3
5	PTC4
6	PTC5
7	PTC6
8	PTC7
9	+VDIO (3.3v)
10	GND

Port C

Pin No.	Description
1	PTD0
2	PTD 1
3	PTD 2
4	PTD 3
5	PTD 4
6	PTD 5
7	PTD 6
8	PTD 7
9	+VDIO (3.3v)
10	GND

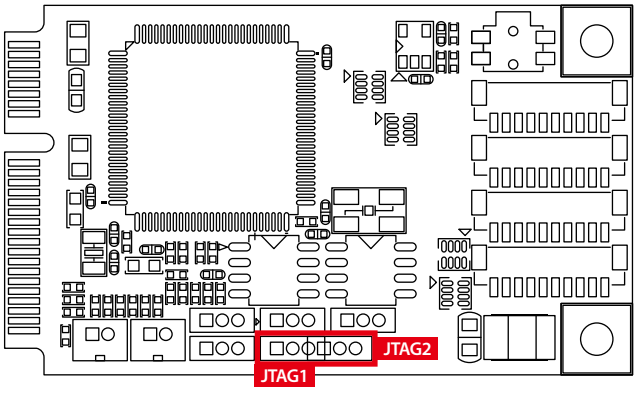
Port D

Pin No.	Description
1	PTE2
2	PTE3
3	PTE4
4	PTE5
5	PTE6
6	PTE24
7	PTE25
8	PTE26
9	+VDIO (3.3v)
10	GND

2.2.2 JTAG Header

This device has extensive debug capabilities including run control, FW upgrade and tracing capabilities.

The standard ARM debug port supports SWD/JTAG interface. Also the JTAG interface is supported on this device.



JTAG1 definition :

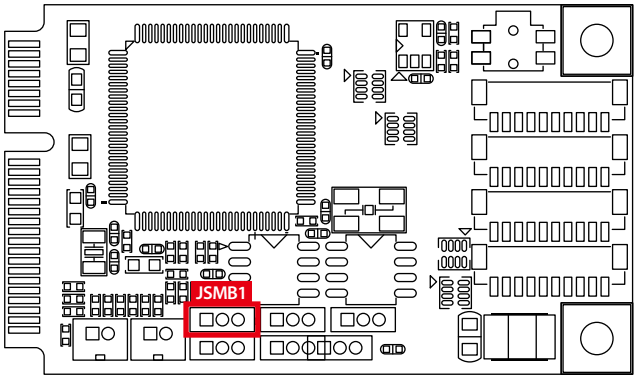
Pin No.	Description
1	JTAG_TMS
2	+V3P3
3	GND

JTAG2 definition :

Pin No.	Description
1	JTAG_TCLK
2	JTAG_TDI
3	JTAG_TDO

2.2.3 System Management Bus (SMBus) Specification

- Standard, Fast, Fast+ and Ultra Fast modes are supported
- HS-mode supported in slave mode
- Multi-master support including synchronization and arbitration
- General call, 7-bit and 10-bit addressing
- Software reset, START byte and Device ID require software support
- For master mode : command/transmit FIFO of 4 words, receive FIFO of 4 words
- For slave mode : separate I2C slave registers to minimize software overhead due to master/slave switching
- support for 7-bit or 10-bit addressing, address range, SMBus alert and general call address
- transmit/receive data register supporting interrupt or DMA requests

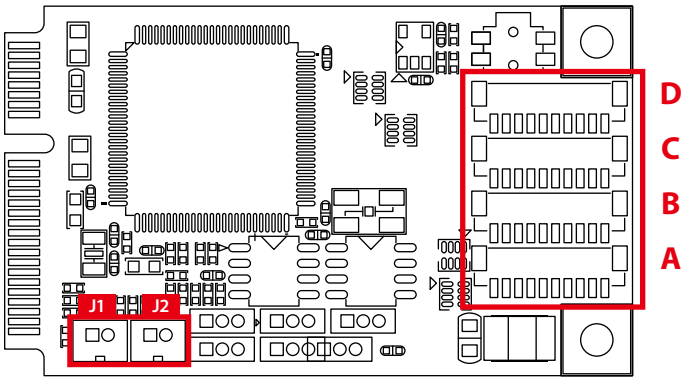


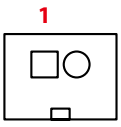
JSMB1 Header definition :

Pin No.	Definition
1	SMB_CLK
2	SMB_DAT
3	GND

2.2.4 Sink/Source Mode Control

The header is combined with DMX-100-E for selecting SINK/SOURCE mode by MCU. (The DMX-100-M can combine with two DMX-100-E cards)
The two headers definition :





J1 is combined with DMX-100-E for selecting SINK/SOURCE mode by MCU.
Note : Group 1 → J1, Port A, Port B

PIN1	PIN2	Definition
0	1	SOURCE
1	0	SINK

J2 is combined with DMX-100-E for selecting SINK/SOURCE mode by MCU.
Note : Group 2 → J2, Port C, Port D

PIN1	PIN2	Definition
0	1	SOURCE
1	0	SINK

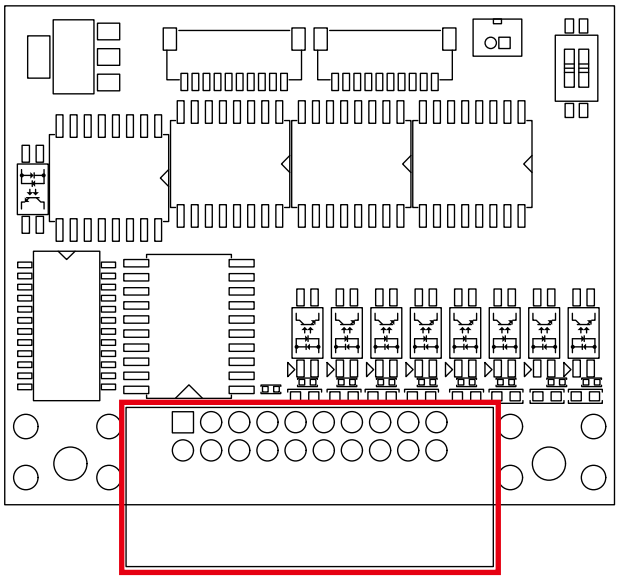
2.3 DMX-100-E I/O Function

There is a 16-bit Isolated DIO that 8 DI/8 DO. DI/DO support NPN (sink) and PNP (Source).

2.3.1 Isolated DIO

There is a 16-bit (8-bit DI, 8-bit DO) connectors in the rear side. DI/DIO support NPN (sink) and PNP (Source) mode, Each DI pin is equipped with a photocoupler for isolated protection. Each DO pin is equipped with isolator function, DO Safety-Related Certifications :

- 4242-VPK Basic Isolation per DIN V VDE V 0884-10 and DIN EN 61010-1
- 3-KVRMS Isolation for 1 minute per UL 1577
- CSA Component Acceptance Notice 5A, IEC 60950-1 and IEC 61010-1 End Equipment Standards
- GB4943.1-2011 CQC Certified

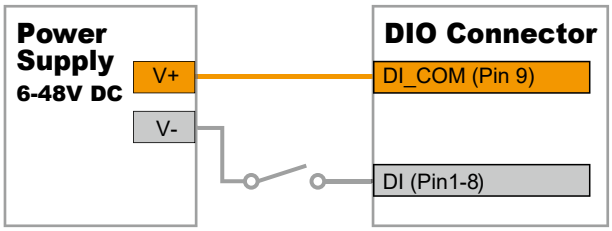


DIO Connectors pin out :

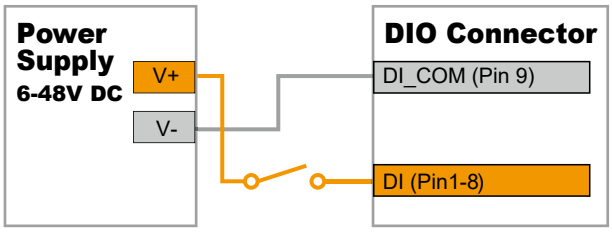
Pin No.	Definition	Pin No.	Definition
1	EXT_IN0	11	EXT_OUT0
2	EXT_IN 1	12	EXT_OUT 1
3	EXT_IN 2	13	EXT_OUT 2
4	EXT_IN 3	14	EXT_OUT 3
5	EXT_IN 4	15	EXT_OUT 4
6	EXT_IN 5	16	EXT_OUT 5
7	EXT_IN 6	17	EXT_OUT 6
8	EXT_IN 7	18	EXT_OUT 7
9	+VDI_COM	19	GND
10	GND	20	+VDIO_EXT

DI reference circuit :

Sink Mode (NPN)

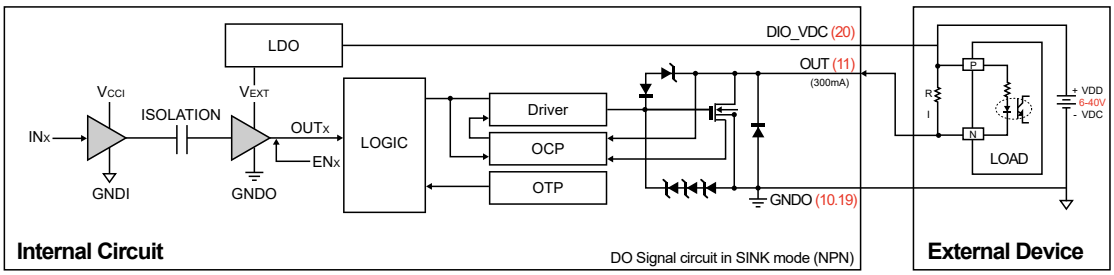


Source Mode (PNP)

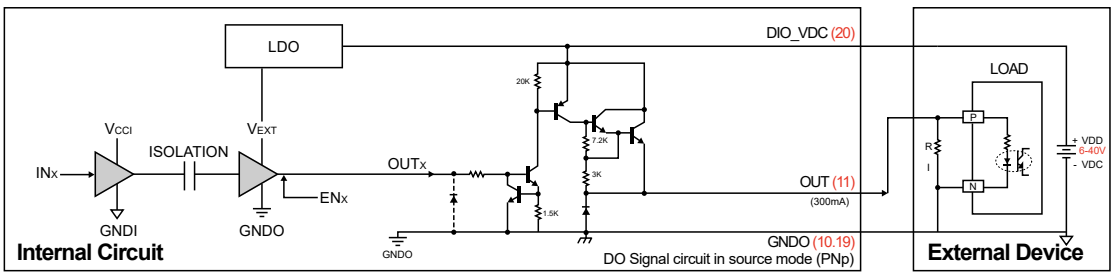


DO reference circuit :

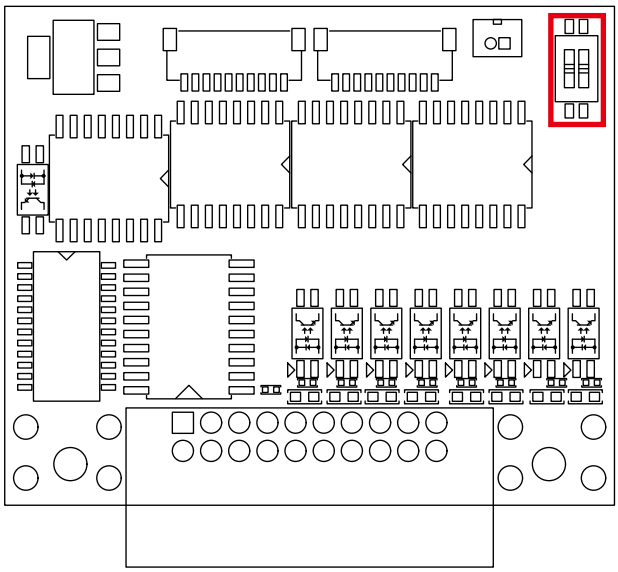
Sink Mode (NPN, Default)

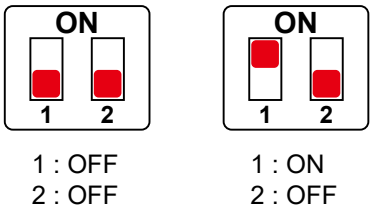


Source (PNP)



2.3.2 SINK/SOURCE Mode control by HW DIP switch





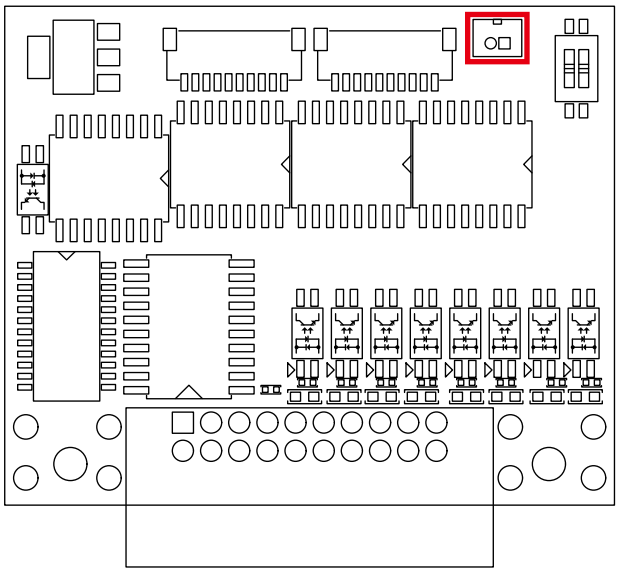
1 : OFF
2 : OFF

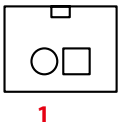
1 : ON
2 : OFF

DIP Switch

PIN1	PIN2	Definition
OFF	ON	SINK
ON	OFF	SOURCE
OFF	OFF	Control by SW (Default)

2.3.3 SINK/SOURCE Mode control by SW



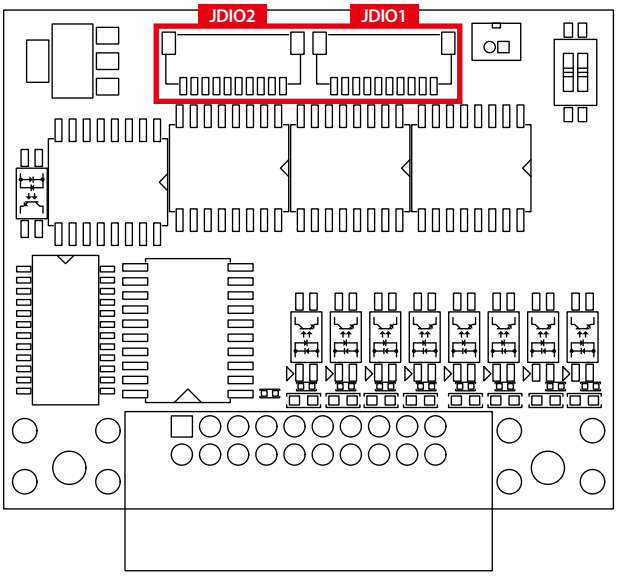


J1 is combined with DMX-100-M J1 or J2 for selecting SINK/SOURCE mode by MCU.

PIN1	PIN2	Definition
0	1	SOURCE
1	0	SINK

2.3.4 IO Connection

There are sixteen programmable I/O (8 Input/8 output) from DMX-100-M. Connect with DMX-100-M Group A/B or Group C/D.



JDIO1

Pin No.	Description
1	DI0
2	DI1
3	DI2
4	DI3
5	DI4
6	DI5
7	DI6
8	DI7
9	+VIN
10	GND

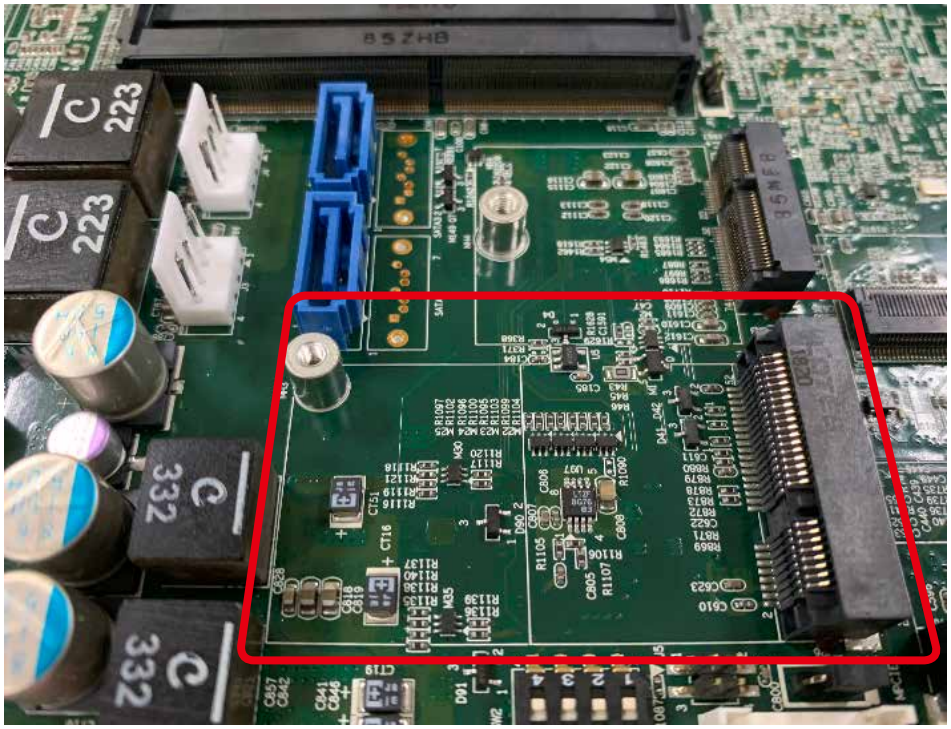
JDIO2

Pin No.	Description
1	DO0
2	DO1
3	DO2
4	DO3
5	DO4
6	DO5
7	DO6
8	DO 7
9	+VIN
10	GND

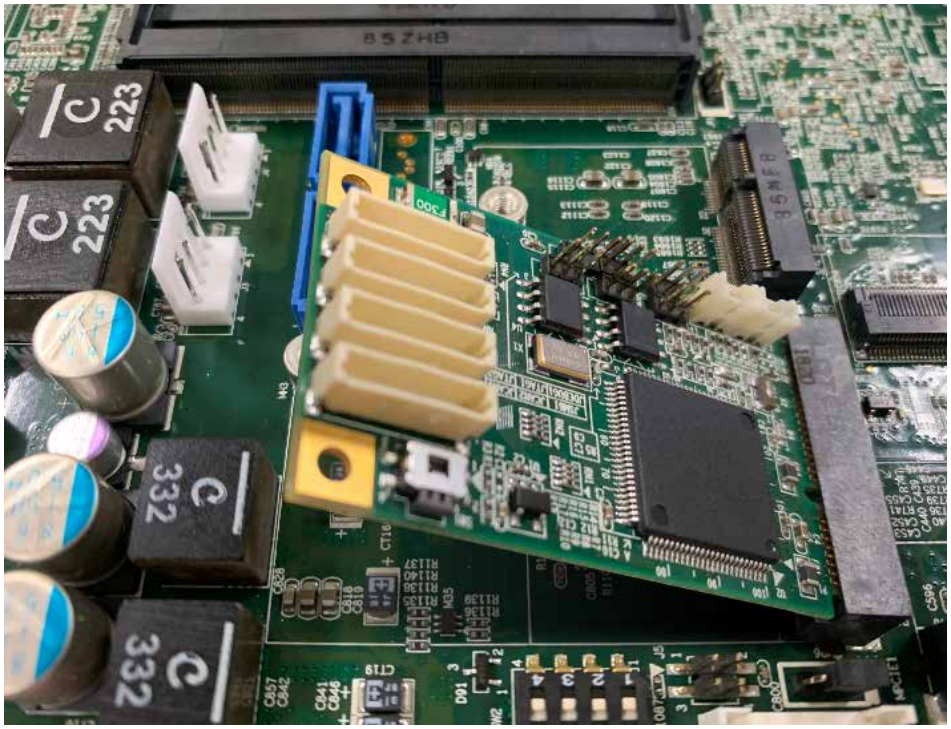
2.4 Installation

2.4.1 Installing DMX-100-M

Step 1 Mini PCIe socket.



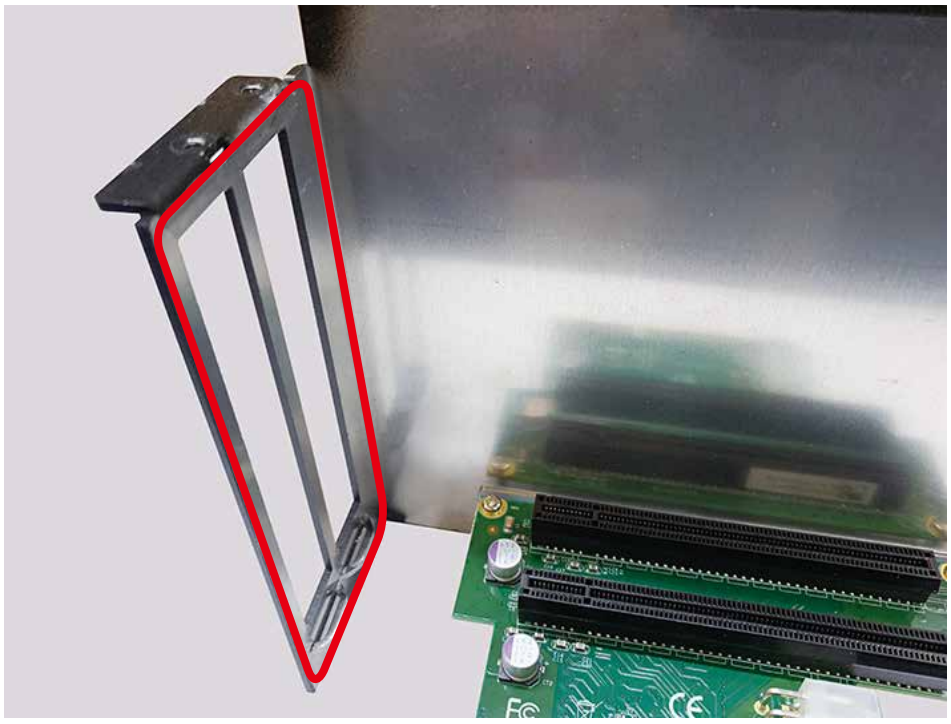
Step 2 Install DMX-100-M.



Step 3 Fasten one PHILLIPS M2.5 screw.

2.4.2 Installing DMX-100

Step 1 PCI standard slot.



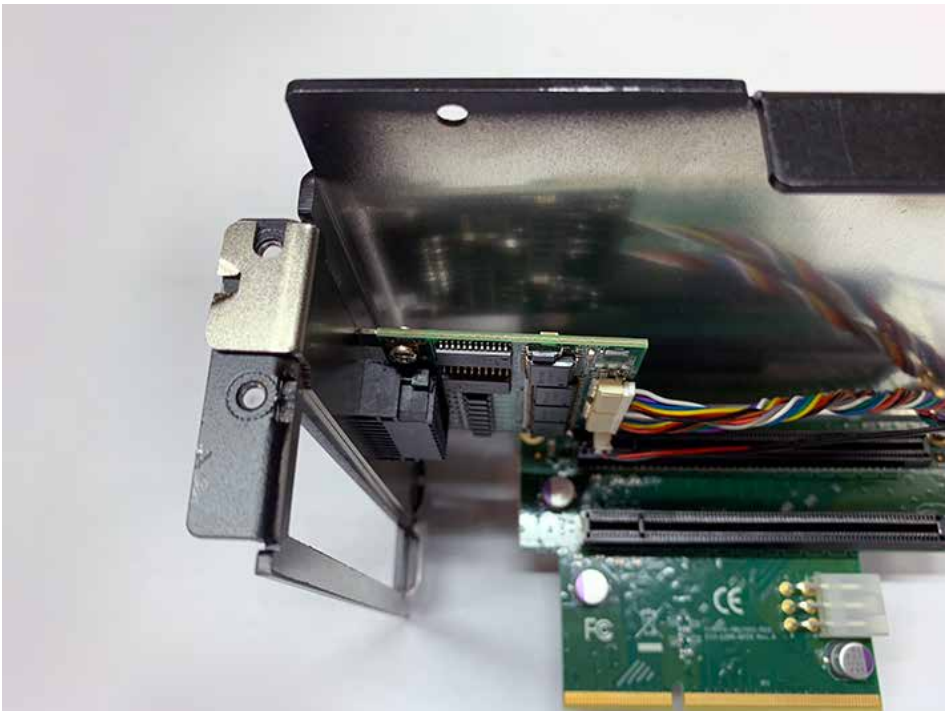
Step 2 Install DMX-100 bracket with D-sub connector to slot.

Step 3 Fasten one PHILLIPS HEX #6-32 screw.

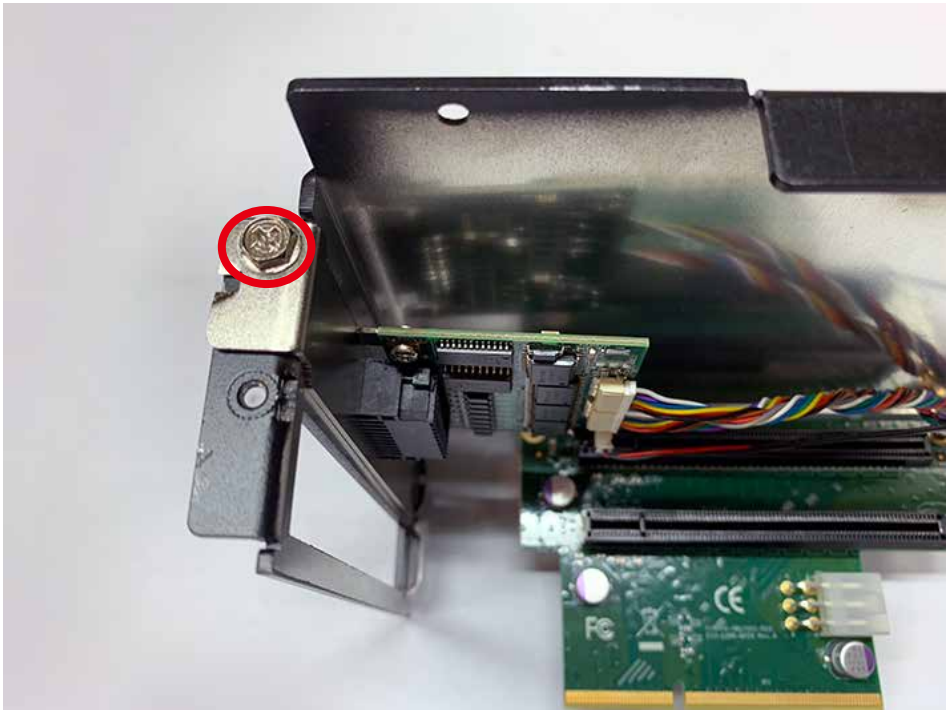
2.4.3 Installing DMX-110

Step 1 PCI standard slot.

Step 2 Install DMX-110 bracket with DMX-100-E to slot.



Step 3 Fasten one PHILLIPS HEX #6-32 screw.



3

DMX-100 SIGNAL CONNECTIONS

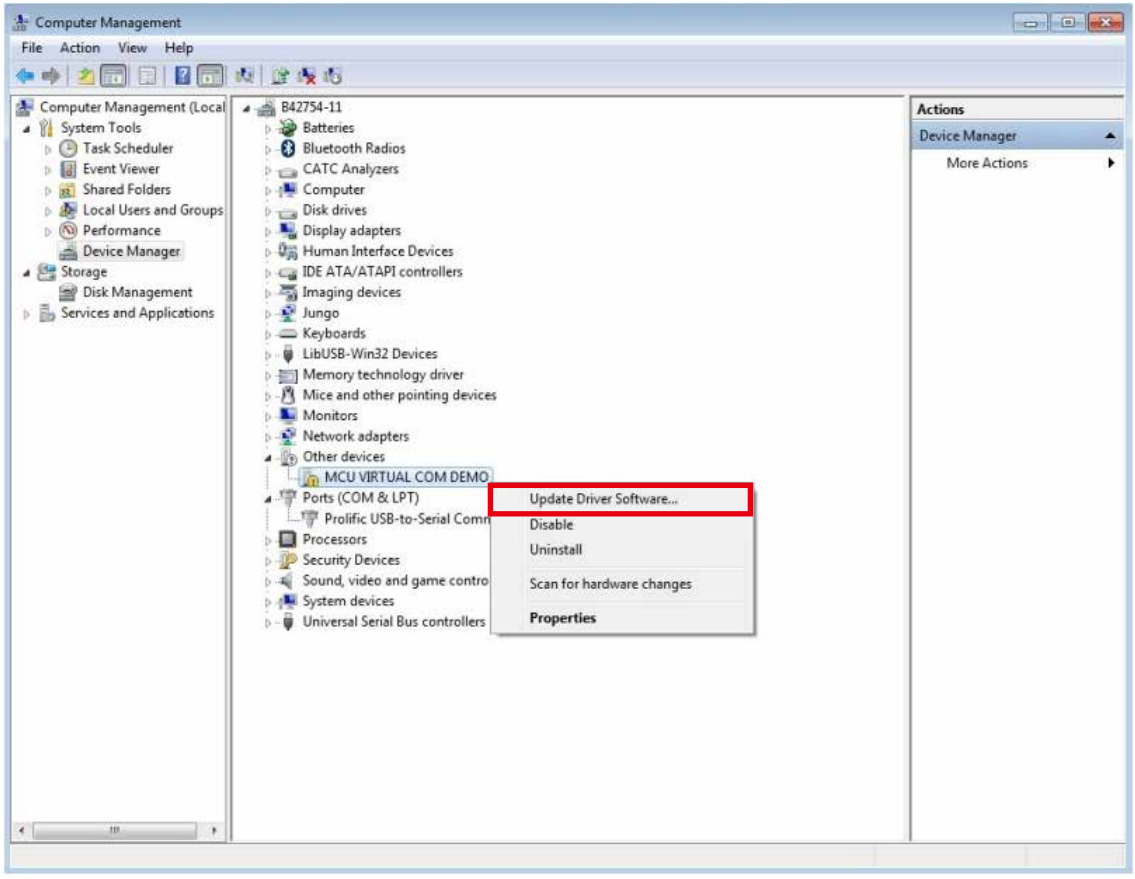
3.1 OS Installation

3.1.1 Window OS Installation

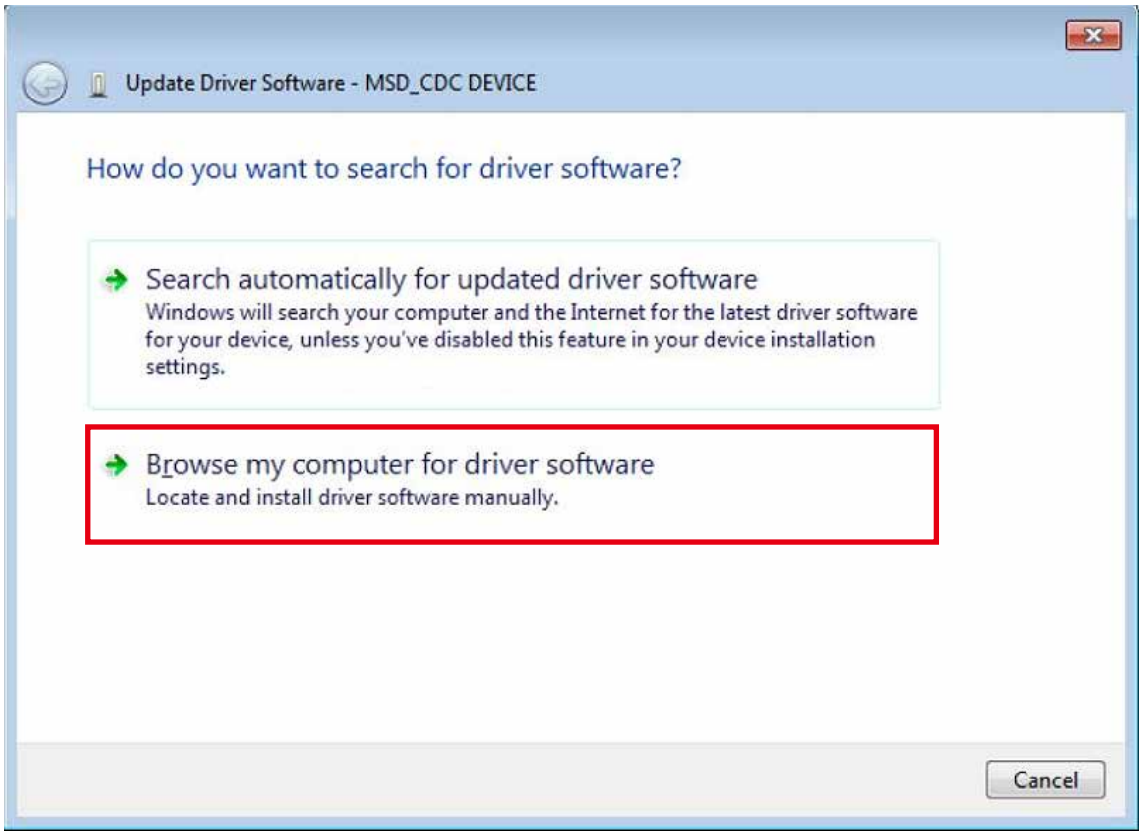
Vecow provide a package which will enable access to the board function though an easy to use API set. The package includes CDC driver and vc_redist_xxx (xxx=x86, x64) plugin package and is available in Windows x86 and x64 version. User must install vc_redist_xxx plugin package before installing the driver.

Below are the steps to install the CDC driver on Windows 7. Use inf folder which we provide.

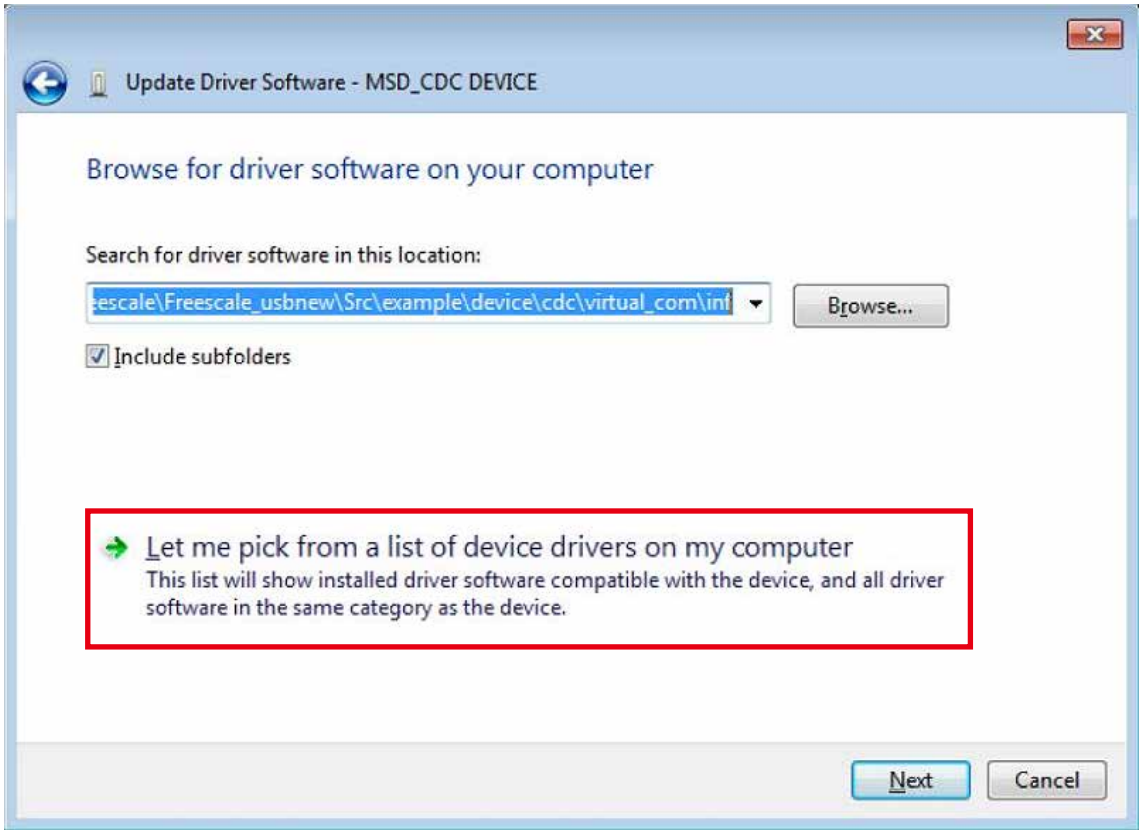
Step 1 Please click “Update Driver Software…”



Step 2 Please chooses “Browse…”

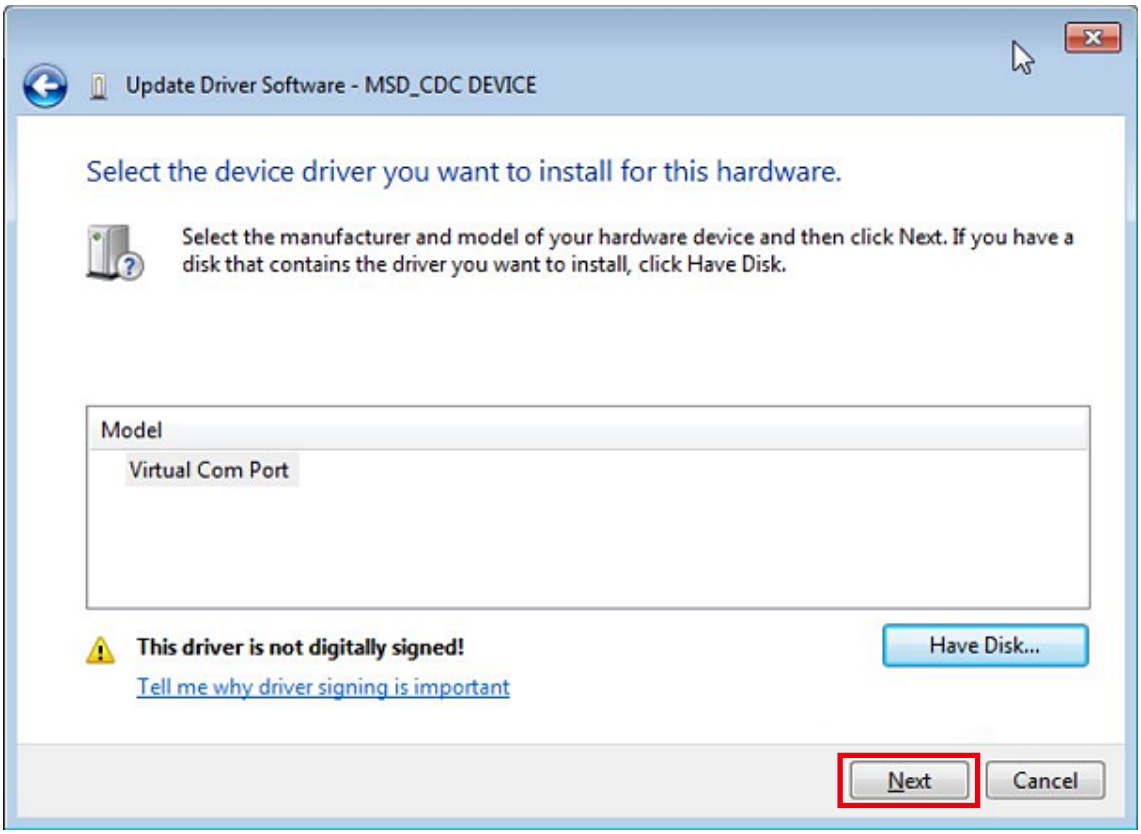


Step 3 Please select “Let me pick…”

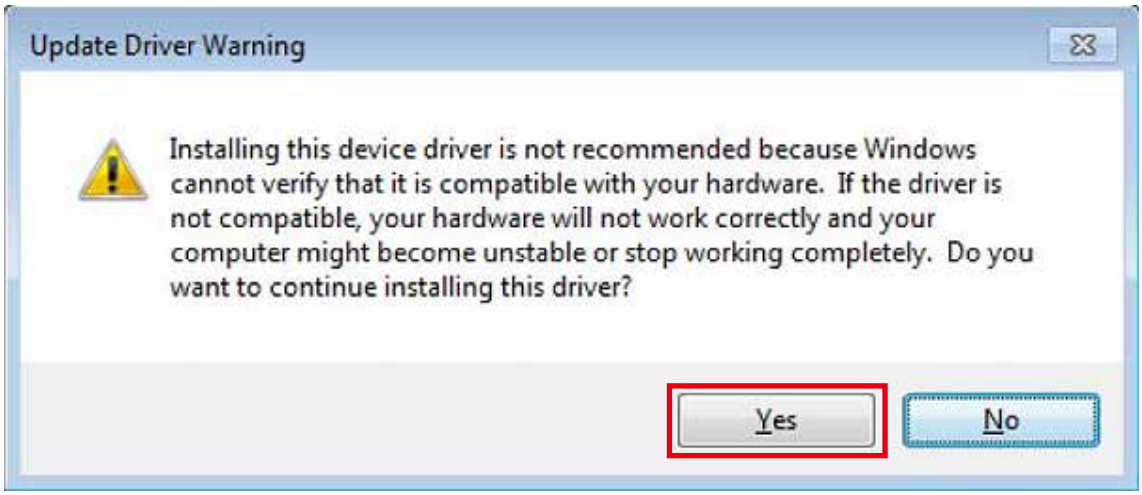


Step 4 Please navigate to your CDC driver location.
<install_dir> \windowsDIO\inf

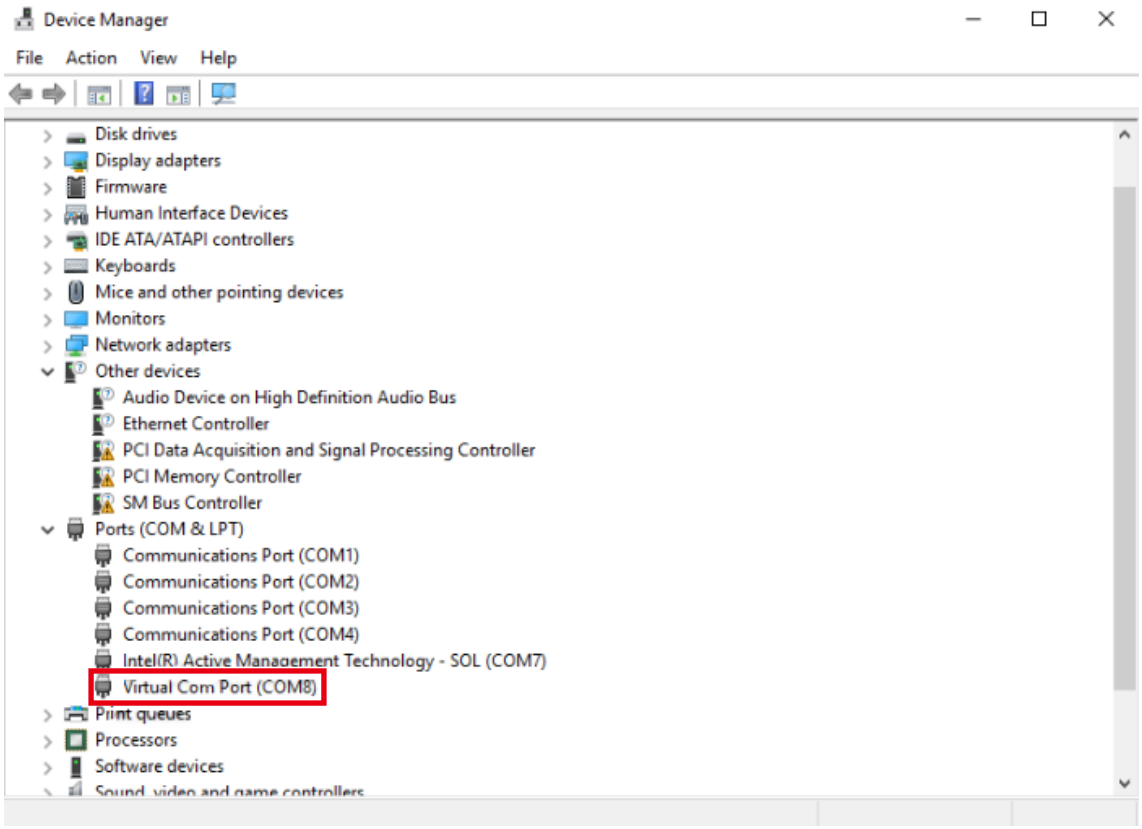
Step 5 Please press “Next”.



Step 6 Please ignore the warning and press “Yes”

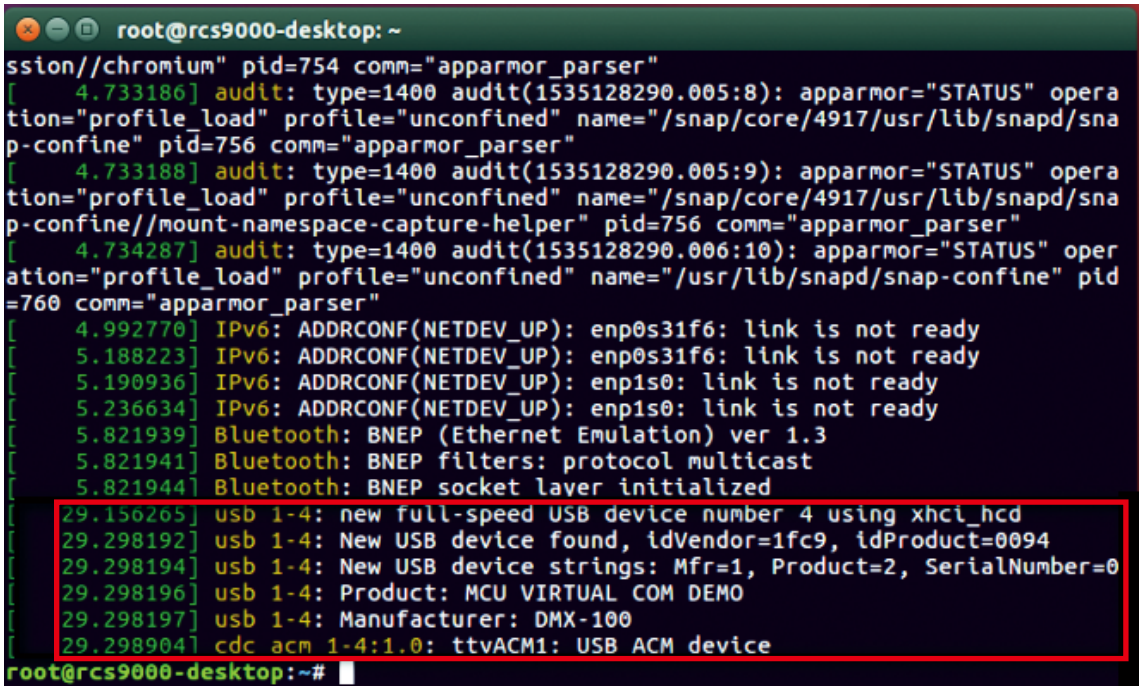


Step 7 Now the CDC driver installed successfully



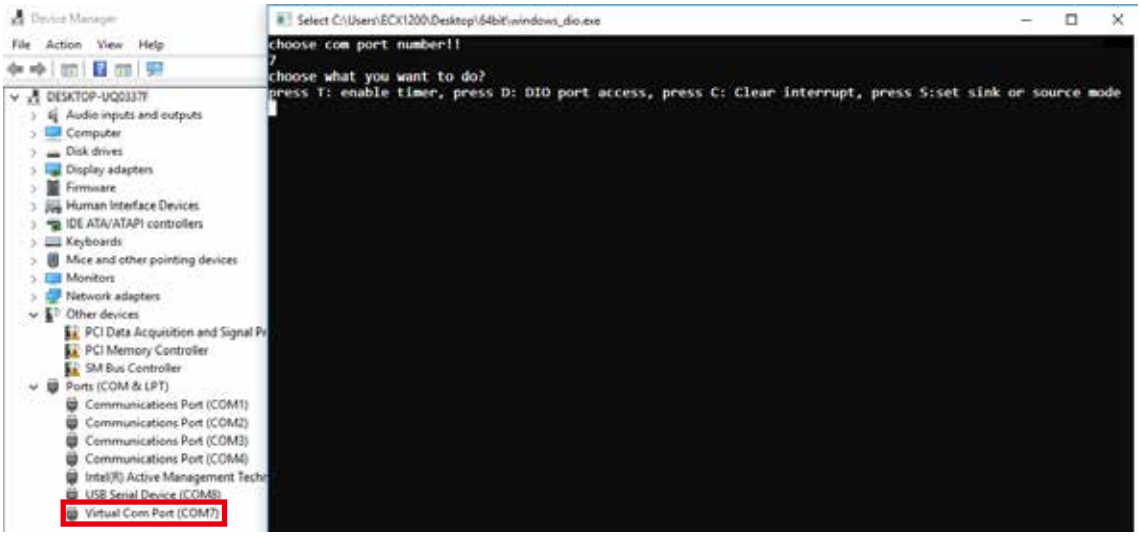
3.1.2 Linux OS Installation

When install DMX-100, the device will be recognized as ttyACMx (x=0,1,2…) by using native CDC-ACM driver. Type command “dmesg” in Linux console to see messages below. In general, the device name would be ttyACM0 or ttyACM1 in Linux.



3.2 Utility

You can use windows_dio.exe to test DMX-100 in Windows. First please choose an available COM Port number.



3.2.1 Function DIO port access

3.2.1.1 Function Input mode

Set port in input mode.

```
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
d
Choose Port[a/b/c/d]:
a
Mode[<S>et/ <G>et]:
s
Direction[<I>n/<O>ut/<H>andler/<C>ounter]:
i
write successful
```

3.2.1.2 Function Output mode

Set port in output mode.

```
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
d
Choose Port[a/b/c/d]:
a
Mode[<S>et/ <G>et]:
s
Direction[<I>n/<O>ut/<H>andler/<C>ounter]:
o
Set Port Status[00 ~ FF]:
ff
write successful
```

3.2.1.3 Function Handler IRQ mode

Set port in interrupt mode, and user can choose rising or falling edge.

```
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
d
Choose Port[a/b/c/d]:
a
Mode[<S>et/ <G>et]:
s
Direction[<I>n/<O>ut/<H>andler/<C>ounter]:
h
Set Interrupt[ <RI>se/<FA>ll ]:
ri
write successful
```

Port A occurs interrupt.

```
choose what you want to do?
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
DMX-100 responses: H, 0x1
```

3.2.1.4 Function Counter mode

Set the number of interrupt which the console will display the message as it reaches to. When reaching to the number of interrupt set by the user, it will show interrupt message. User can disable counter mode by setting counter '0'. Counter is a 32-bit register.

```
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
d
Choose Port[a/b/c/d]:
a
Mode[<S>et/ <G>et]:
s
Direction[<I>n/<O>ut/<H>andler/<C>ounter]:
c
Set Interrupt[ <RI>se/<FA>ll ]:
fa
set your counter
2
write successful
```

When counter occurs, it will show 4 bits of counter buffer.
0x1=>0001=>port A irq
0x2=>0010=>port B irq
0x4=>0100=>port C irq
0x8=>1000=>port D irq

```
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
DMX-100 responses: C, 0x1
```

3.2.1.5 Function Get status

It will get port status.

```
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
d
Choose Port[a/b/c/d]:
a
Mode[<S>et/ <G>et]:
g
write successful
choose what you want to do?
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
DMX-100 responses: G, 0xfe
```

3.2.2 Function Timer mode

User can input time. When the time is out, microcontroller will response each port status. Timer will keep report status, and store the result in 32-bit register.

```
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
t
set your timer 0~4294967296 seconds
30
write successful
```

```
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
DMX-100 responses: T, 0x0 0x24 0xf7 0xff
```

When you want to disable timer, just choose timer again and input 0 seconds.

3.2.3 Function Clear interrupt

It will clear interrupt buffer. When using “clear interrupt”, it will clear all ports of interrupt buffer. The examples are shown below. First picture shows port A and port B occurring interrupt. Second picture shows the result of using clear interrupt and then letting Port B get interrupt again.

```
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
DMX-100 responses: H, 0x3
```

```
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
DMX-100 responses: H, 0x2
```

When interrupt occurs, it will show 4 bits of interrupt buffer.
0x1=>0001=>port A irq
0x2=>0010=>port B irq
0x4=>0100=>port C irq
0x8=>1000=>port D irq

3.2.4 Function Sink or Source mode

Set DIO becomes sink or source mode.

```
press T: enable timer, press D: DIO port access, press C: Clear interrupt, press
 S:set sink or source mode
s
choose DIO1 for port A(DI) and B(DO) press(1) or DIO2 for port C(DI) and D(DO) p
ress(2)
1
Set Sink or Source mode[(SI)nk / (SO)urce / (DI)sable]:
si
if you want set GPIO again, you should (DI)sable DIO function
write successful
```

DIO1 for port A and B, DIO2 for port C and D. Port A and C becomes input pin, B and D becomes output pin.
Makes device become GPIO again, please press <DI>sable DIO.

A

APPENDIX A : Software API

Vecow provides API and data structure (DIO_Dll.h) for user programming in DMX-100. API is based on a dynamic library (DLL) in Windows and static library (.a) in Linux. The only different API in Linux and Windows is OpenCom().

OpenCom (Windows)

Description: Open virtual COM port

Syntax: `int OpenCom(int com)`

Parameter: input virtual com number

Return value: -1 open virtual com port fail, 0 open virtual com port successfully.

OpenCom (Linux)

Description: Open virtual COM port. In Linux, user can use ShowCom() API to show which device name ttyACMx (x=1,2,3…) can be used. **After choose device name, call OpenCom() API.**

Syntax: `int OpenCom(char*str)`

Parameter: input virtual com port device name (/dev/ttyACMx)

Return value: -1 open virtual com port fail, 0 open virtual com port successfully.

ShowCom (Linux)

Description: It shows how many DMX-100 virtual COM Ports in your PC. After calling ShowCom() API, it will return device name and name length.

Syntax: `void ShowCom(char**nameString, int*nameSize)`

Parameter: nameString is the device name of USB to URAT. If device has more than one DMX-100, it will return (ex.ttyACM1+ttyACM2+….).
nameSize is the total length of nameString.

ComIO

Description: Send user input structure about I/O direction and status to microcontroller.

Syntax: void ComIO(PACKET_INFO packet_info)

PACKET_INFO structure:

```
typedef struct
{
	char v_com;// a b c d
	char mode;// set or get s/g
	char dir_set;// set pin Input/Output
	char status_set[3];//0x00~0xff
	uint8_t rtn_ack;
}PACKET_INFO;
```

Parameter:
Vcom: a b c d port
Mode: set or get port status
Dir_set: input or output or interrupt or counter mode
Satus_set[3]: set port status always 0, except output mode, user can set value 00~FF.
Rtn_ack: for expansion function, only the clear function set to 1, the rest of them set 0.

TimesIO

Description: Send timer or counter input to microcontroller.

Syntax: void TimesIO(PACKET_TIMES packet_info)

PACKET_TIMES structure:

```
typedef struct
{
	PACKET_INFO packet_times;
	uint32_t times;
}PACKET_TIMES;
```

Member: (input parameter)
PACKET_INFO structure.
Times: is timer or counter.

VccIO

Description: Set DIO in source or sink mode

Syntax: void VccIO(DIO vcc_dio)

DIO structure:
```
typedef struct
{
	unsigned char cardNum;
	unsigned char sink;
	unsigned char source;
}DIO;
```

Member: (input parameter)
cardNum: choose card 1 or 2.
Sink and source: if sink mode set ‘SI’, source mode ‘SO’, disable set ‘DI’.

CBregister

Description: A register callback function which will receive microcontroller responds.

Syntax: void CBregister(cbfunction callback)

Responds packet structure:
```
typedef struct
{
	unsigned char status;
	unsigned char res;
	unsigned char res1;
	unsigned char res2;
	unsigned char res3;
}RX_PACKET;
```

Parameter:
Status: G(get command), H(handler command), T(timer command), C(counter command)
Res~res3: microcontroller responds status.

CloseCom

Description: Close com port

Syntax: void CloseCom()

Vecow

For further support information, please visit www.vecow.com