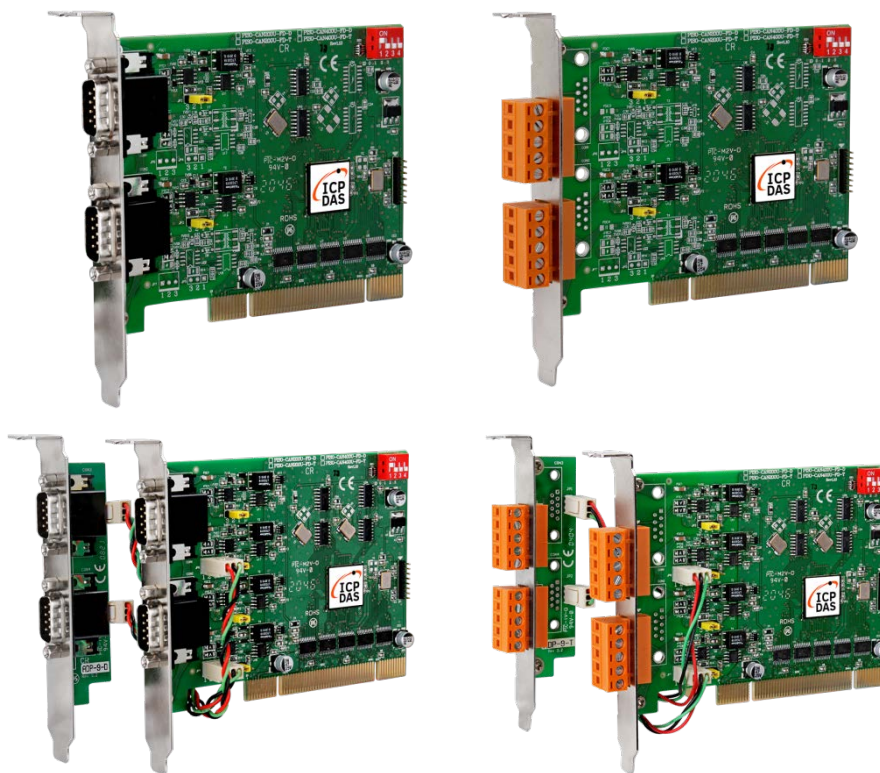


PISO-CANFD Series User Manual

Version 1.0.0, Feb. 2021



Service and usage information for

PISO-CAN200U-FD-D / PISO-CAN200U-FD-T

PISO-CAN400U-FD-D / PISO-CAN400U-FD-T

Warranty

All products manufactured by ICP DAS are under warranty regarding defective materials for a period of one year, beginning from the date of delivery to the original purchaser.

Warning

ICP DAS assumes no liability for any damage resulting from the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Copyright © 2021 by ICP DAS Co., Ltd. All rights are reserved.

Trademarks

Names are used for identification only and may be registered trademarks of their respective companies.

Contact us

If you encounter any problems while operating your device, please feel free to contact us by email at: service@icpdas.com.

Table of Contents

1.	<i>Introduction</i>	<i>5</i>
1.1.	Specifications.....	6
1.2.	Features.....	7
1.3.	Overview	8
1.3.1.	Board Layout	8
1.3.2.	Pin Assignments	10
1.3.3.	Board Number Switch Settings	11
1.3.4.	Terminal Resistor Jumper Settings	12
2.	<i>Getting Started</i>	<i>13</i>
2.1.	Installing the Hardware	13
2.2.	Installing the Windows Driver	14
2.3.	Wiring Connections.....	17
2.4.	Software Utility	18
2.4.1.	Connect to the board	19
2.4.2.	Send CAN/CAN FD messages	22
2.4.3.	Receive CAN/CAN FD messages	25
2.4.4.	Check CAN Bus Status	28
3.	<i>Windows API Function Reference.....</i>	<i>31</i>
3.1.	API Library Overview	31
3.2.	API Library Function Table	32
3.3.	API Library Flow Diagram	35
3.4.	System Information API	36
3.1.1.	CANFD_GetDllVersion.....	36
3.1.2.	CANFD_GetBoardInf	37
3.1.3.	CANFD_TotalBoard.....	39
3.1.4.	CANFD_GetCardBoardSwitchNo.....	40
3.1.5.	CANFD_GetCardBoardID.....	41
3.1.6.	CANFD_GetCardPortNum	42
3.1.7.	CANFD_GetCardFPGA FWVer	43

3.1.8.	CANFD_ActiveBoard.....	44
3.1.9.	CANFD_CloseBoard	45
3.1.10.	CANFD_BoardIsActive	46
3.5.	CAN Bus API.....	47
3.2.1.	CANFD_Reset.....	47
3.2.2.	CANFD_Init	48
3.2.3.	CANFD_SetBitRate.....	49
3.2.5.	CANFD_SetBitRateWithSP	51
3.2.6.	CANFD_GetBitRate.....	53
3.2.7.	CANFD_GetBitRateWithSP	55
3.2.8.	CANFD_SetFilterAllPass.....	57
3.2.9.	CANFD_SetFilterFormat	58
3.2.10.	CANFD_SetFilter	59
3.2.11.	CANFD_SetOPMode	61
3.2.12.	CANFD_GetOPMode.....	63
3.2.13.	CANFD_SetISOCRCEn	65
3.2.14.	CANFD_GetISOCRCEn	67
3.2.15.	CANFD_SendCANMsg.....	69
3.2.16.	CANFD_RecvCANMsg.....	71
3.2.17.	CANFD_RecvCANMsgCnt	74
3.2.18.	CANFD_SetFIFOStatus.....	75
3.2.19.	CANFD_GetFIFOStatus.....	77
3.2.20.	CANFD_GetCANStatus.....	79
3.2.21.	CANFD_GetBUSDiagnostic	81
3.6.	Error Code Definitions.....	83
4.	<i>Appendix</i>	86
4.1.	Revision History.....	86
4.2.	Dimensions.....	87
4.2.1.	PISO-CAN200U-FD-D / PISO-CAN400U-FD-D	87
4.2.2.	PISO-CAN200U-FD-T / PISO-CAN400U-FD-T	88

1. Introduction

CAN FD (CAN with Flexible Data-Rate) is a newer extension version of the CAN 2.0 protocol. It was developed by Bosch and was released in 2012. It has been significantly improved during the standardization process and is nowadays in ISO 11898-1:2015. The CAN FD speeds up the data transmission and packs more data into each message.

PISO-CANFD series board (included PISO-CAN200U-FD and PISO-CAN400U-FD) is a very powerful and economic solution for an active CAN board, containing two/four CAN channels that cover a wide range of CAN applications. It uses Microchip CAN FD controllers and TI TCAN1042HG series transceivers, which provide bus arbitration and error detection features, combined with auto-correction and re-transmission functionality. As the PISO-CANFD series board is state-of-the-art, it can be installed in either a Universal PCI bus.



1.1. Specifications

Model	PISO-CAN200U-FD-D	PISO-CAN400U-FD-D	PISO-CAN200U-FD-T	PISO-CAN400U-FD-T
PC Bus				
Type	Universal PCI, 3.3 V and 5 V, 33 MHz, 32-bit, plug and play			
Board No.	By DIP switch			
CAN Interface				
Controller	Microchip MCP2518FD			
Transceiver	TI TCAN1042HG			
Ports	2	4	2	4
Connector	9-pin Male D-Sub		5-pin screw terminal block	
Baud Rate	CAN bit rates: 10 ~ 1000 kbps, CAN FD bit rates for data field: 100 ~ 10000 kbps			
Isolation	3000 V _{DC} for DC-to-DC, 3000 V _{rms} for photo-couple			
Terminal Resistor	Jumper for 120 Ω terminator resistor			
Power				
Power Consumption	150 mA @ 5 V	200 mA @ 5 V	150 mA @ 5 V	200 mA @ 5 V
Software				
Drivers	Windows 7/8.1/10 (32-bit/64-bit)			
Library/Demo Languages	C#.Net, VB.Net, VC++.Net			
Mechanical				
Dimensions (mm)	121.7 x 21.6 x 92.7 (W x L x H)			
Environmental				
Operating Temperature	0 to 60 °C			
Storage Temperature	-20 to 70 °C			
Humidity	5 to 85% RH, Non-condensing			

Attention:

The maximum CAN FD data rate can be exceeded depending on the concrete operating conditions (cable length, network topology, settings,...), but it can also not be reached.

1.2. Features

- Compatible with the ISO 11898-2 standard
- Compatible with CAN specification 2.0 A/B and FD
- CAN FD support for ISO and Non-ISO (Bosch) standards switchable
- CAN FD bit rates for data field from 100 kbps to 10 Mbps
- CAN bit rates from 10 kbps to 1000 kbps
- Support CAN Bus message filter configuration
- Board number selectable via DIP switch
- Built-in jumper to select 120 ohm terminal resistor for CAN Bus
- Drivers provided for Windows 7/8.1/10
- Demos and libraries provided for C#.Net, VB.Net and VC++.Net
- 2500 Vrms photo couple isolation on the CAN bus

1.3. Overview

The following is a description of the hardware settings for the PISO-CANFD series board, including the board layout, pin assignments, jumper and switch selection, LED indicators, and the configuration for the wiring connections.

1.3.1. Board Layout

The following is the layout for the PISO-CANFD series board, illustrating the positions of the various connectors, jumpers and switches on the board.

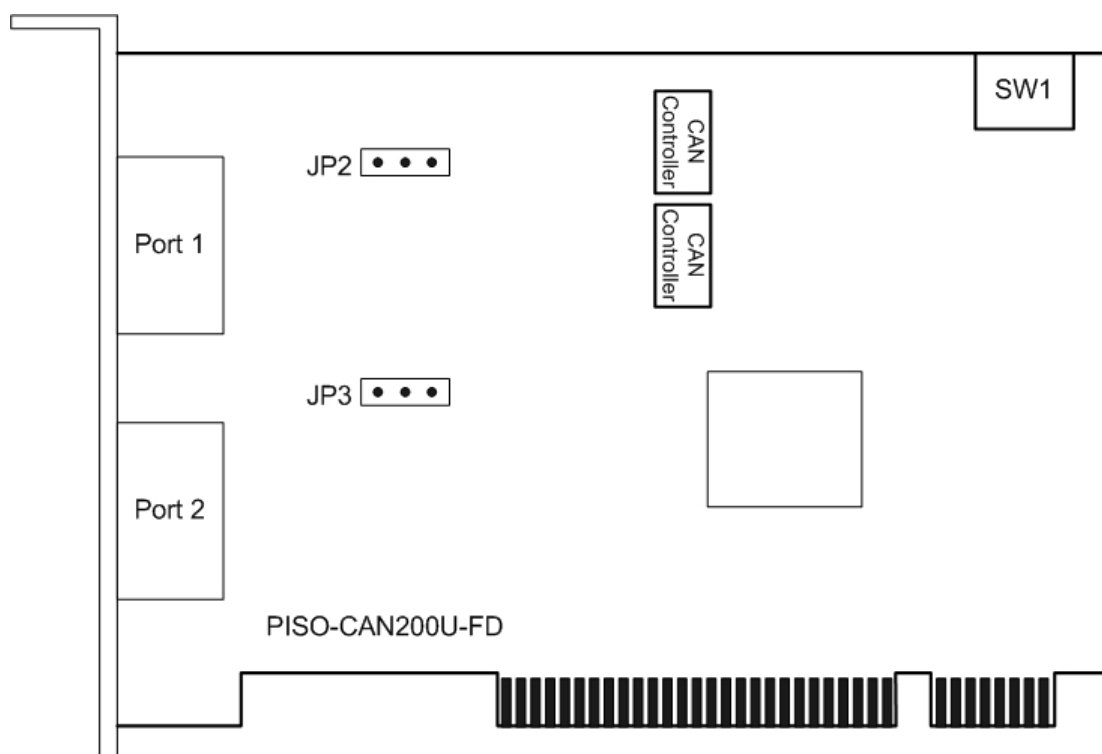


Figure 1.3-1 PISO-CAN200U-FD Board Layout

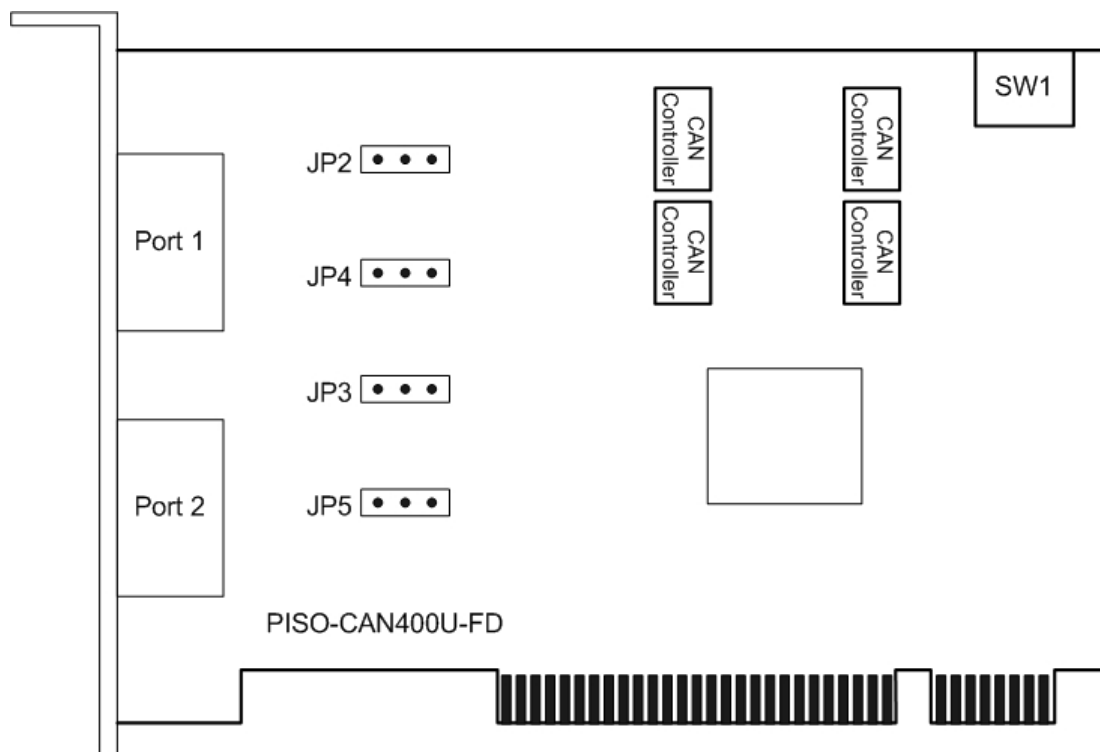


Figure 1.3-2 PISO-CAN400U-FD Board Layout

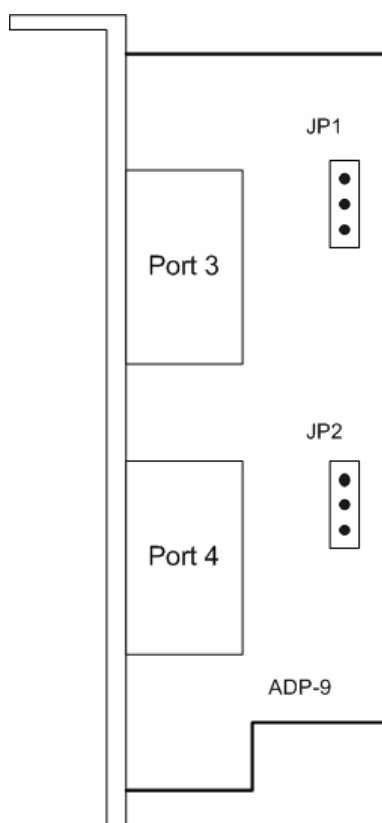
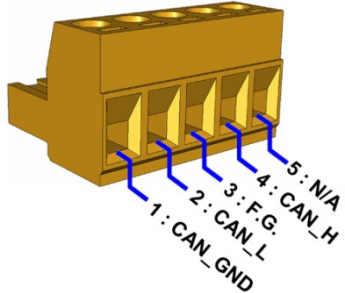


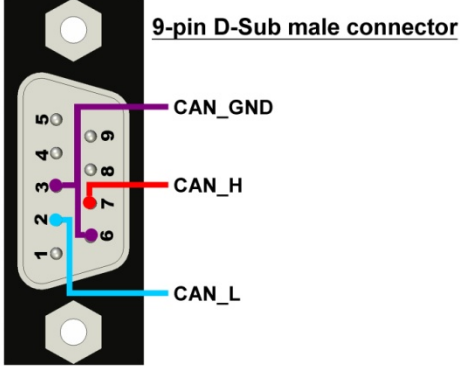
Figure 1.3-3 ADP-9 Board Layout

1.3.2. Pin Assignments

The pin assignments for the 5-pin screw terminal connector and 9-pin Male D-Sub connector on the PISO-CANFD series board are shown below.

Pin Assignments for the 5-pin screw terminal connector			
Pin No.	Name	Description	
1	CAN_GND	CAN_Gnd, signal line for the CAN port.	
2	CAN_L	CAN_Low, signal line for the CAN port.	
3	F.G.	Frame Ground.	
4	CAN_H	CAN_High, signal line for the CAN port.	
5	N/A	Not used	

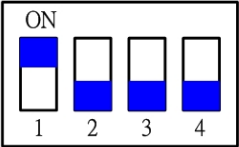
Pin Assignments for the 9-pin Male D-Sub connector		
Pin No.	Name	Description
1	N/A	Not used
2	CAN_L	CAN_Low, signal line for the CAN port.
3	CAN_GND	CAN_Gnd, signal line for the CAN port.
4	N/A	Not used
5	N/A	Not used
6	CAN_GND	CAN_Gnd, signal line for the CAN port.
7	CAN_H	CAN_High, signal line for the CAN port.
8	N/A	Not used
9	N/A	Not used



Electronic circuits are always influenced by different levels of Electrostatic Discharge (ESD), which become worse in a continental climate area. The built-in F.G. provides a path for conducting the ESD to the earth ground. Therefore, connecting the F.G. correctly can enhance the ESD protection capabilities and improve the reliability of the board. Note that wiring of the F.G. is not necessary. The wiring configuration can be modified based on the specific application.

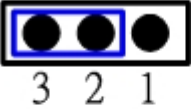
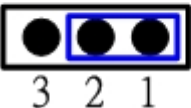

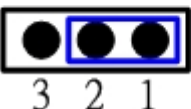

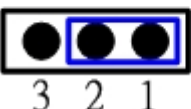

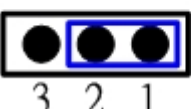
1.3.3. Board Number Switch Settings

The following provides a description of the SW1 DIP switch, which can use to configure board number of the PISO-CANFD series board.

Switch	Description	Status
SW1	SW1 is a DIP switch that is used to configure the board number for the PISO-CANFD series card. For example, if the switch on the left-hand side, (ex., DIP switch 1, is set to ON, as indicated in the figure, the board number will be set to 1. The board number can range from 0 to 15. Note that the board number for each PISO-CANFD series card installed in the Host PC must be unique.	<div><p>DIP Switch</p><p>Thus configuration indicates that the board number is set to 1.</p></div>

1.3.4. Terminal Resistor Jumper Settings

The following provides a description of the resistor jumpers for the CAN Bus terminal resistor, which can be used to enable or disable the terminal resistor of PISO-CANFD series board.

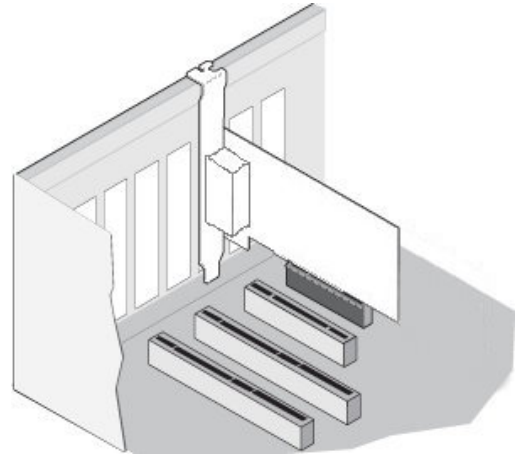
Jumper	Description	Status
JP2	Used to enable or disable the 120Ω terminal resistor for CAN Bus Port1.	JP2  Enabled
		JP2  Disabled
JP3	Used to enable or disable the 120Ω terminal resistor for CAN Bus Port2.	JP3  Enabled
		JP3  Disabled
JP4	Used to enable or disable the 120Ω terminal resistor for CAN Bus Port3.	JP4  Enabled
		JP4  Disabled
JP5	Used to enable or disable the 120Ω terminal resistor for CAN Bus Port4.	JP5  Enabled
		JP5  Disabled

2. Getting Started

This section is a description of how to begin using the PISO-CANFD series board, including installing the hardware, windows driver, making the wiring connections and using software utility and demos to test the board.

2.1. Installing the Hardware

- Step 1:** Shut down and power off the computer.
- Step 2:** Remove all the covers from the computer.
- Step 3:** Select an unused PCI slot.
- Step 4:** Carefully insert the PISO-CANFD series board into the PCI slot and secure the board in place.
- Step 5:** Replace the covers on the computer.
- Step 6:** Reconnect the power supply and power on the computer.
- Step 7:** Once the computer reboots, follow section 2.2 to install the windows driver of PISO-CANFD series board.



2.2. Installing Windows Driver

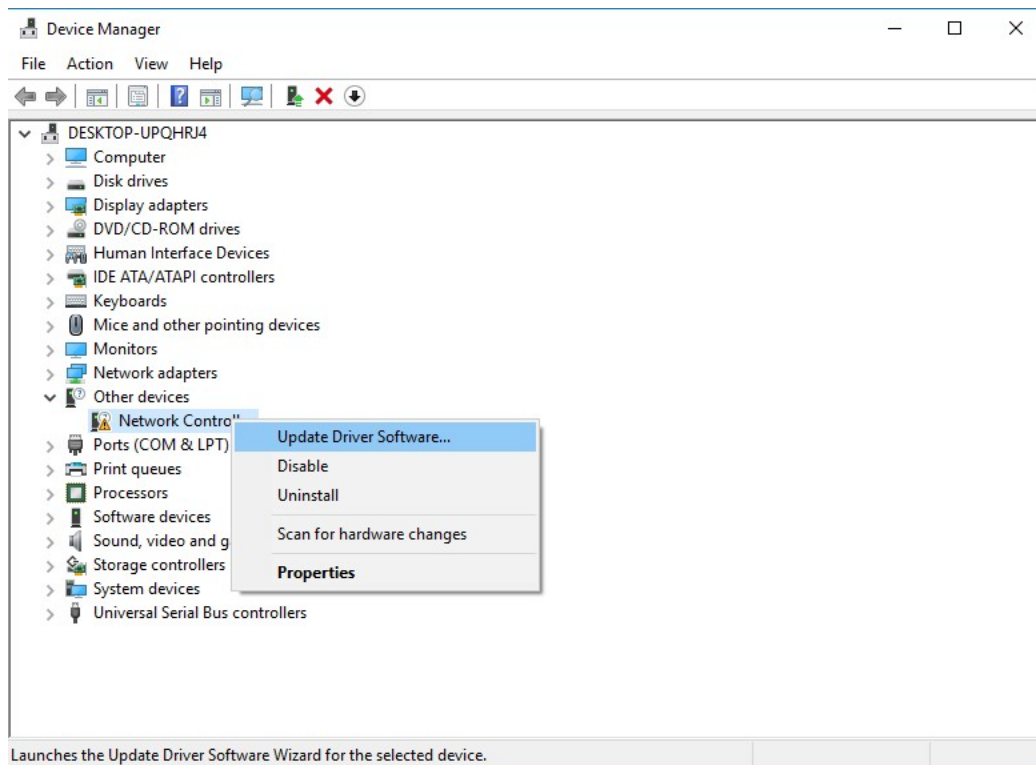
To use the PISO-CANFD series board in a Windows environment, the correct driver for the specific version of the Windows operating system must be installed. The drivers can be downloaded from the ICP DAS website or the following link:

<https://www.icpdas.com/en/download/show.php?num=3200>

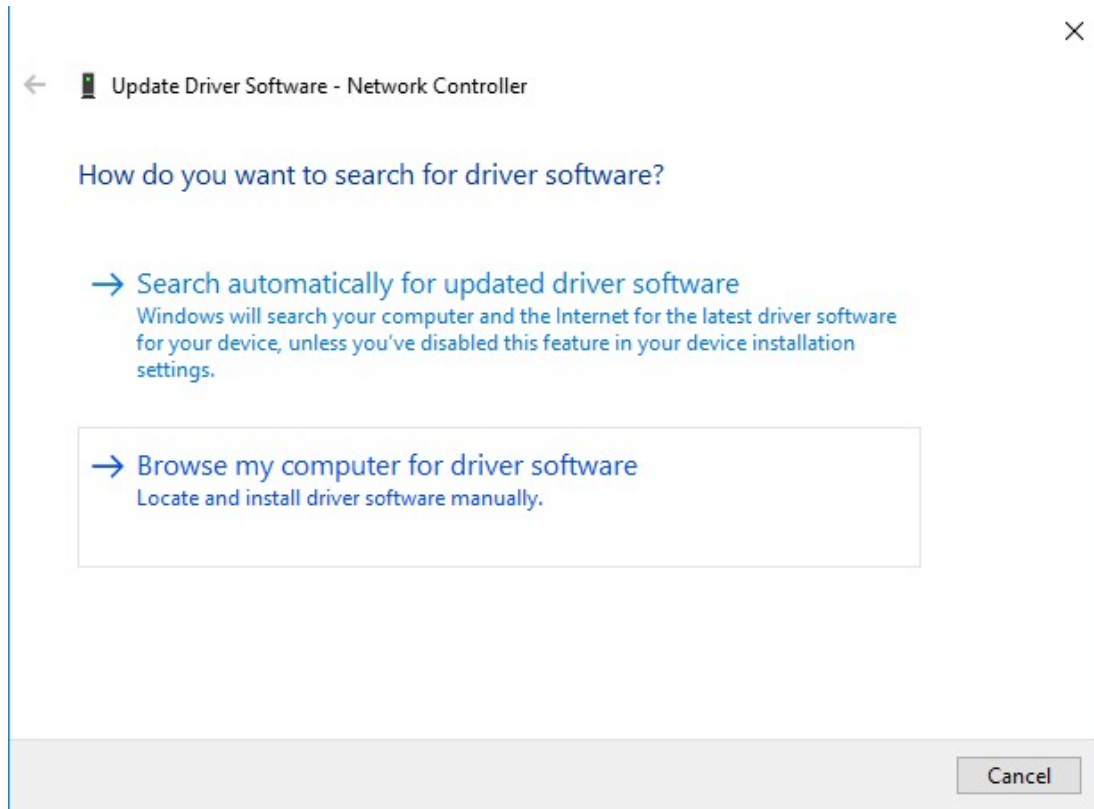
The following is a description of the installation procedure for the Win10 operating system. The installation procedures for other versions of Windows are similar.

Step1: Right-click the **Start** button or press the **Windows Logo + X** key combination on the keyboard and, from the list, click to select **Device Manager**.

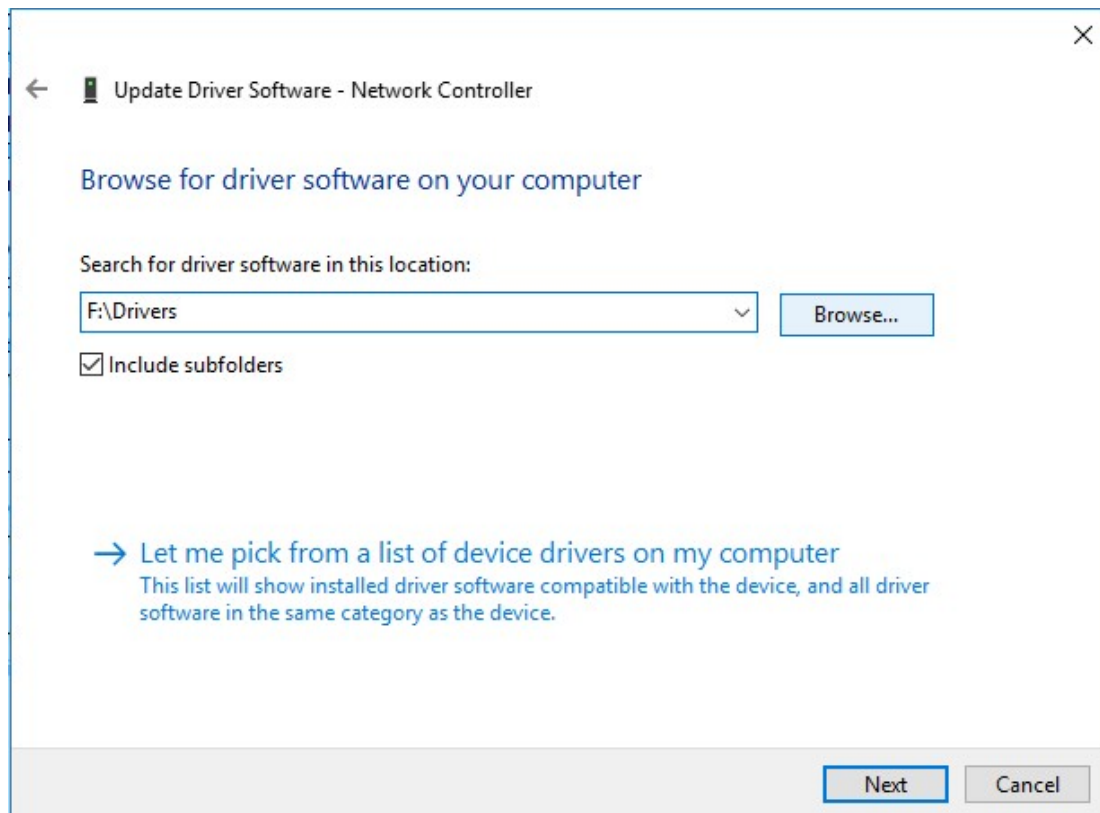
Step 2: On the **Device Manager** screen, select the **Network Controller** device from **Other devices** item, then right-click the mouse button. Then select **Update Driver Software...** to continue.



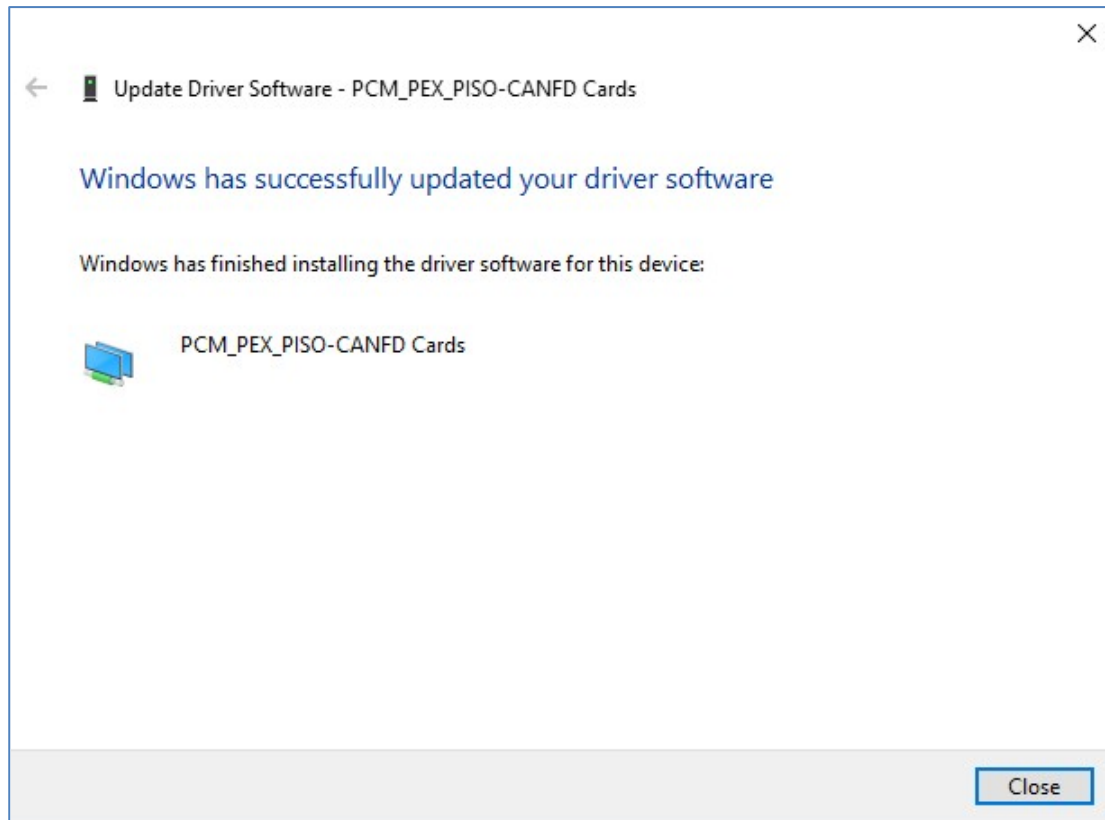
Step 3: On the **Update Driver Software – Network Controller** screen, click the **Browse my computer for driver software** to continue.



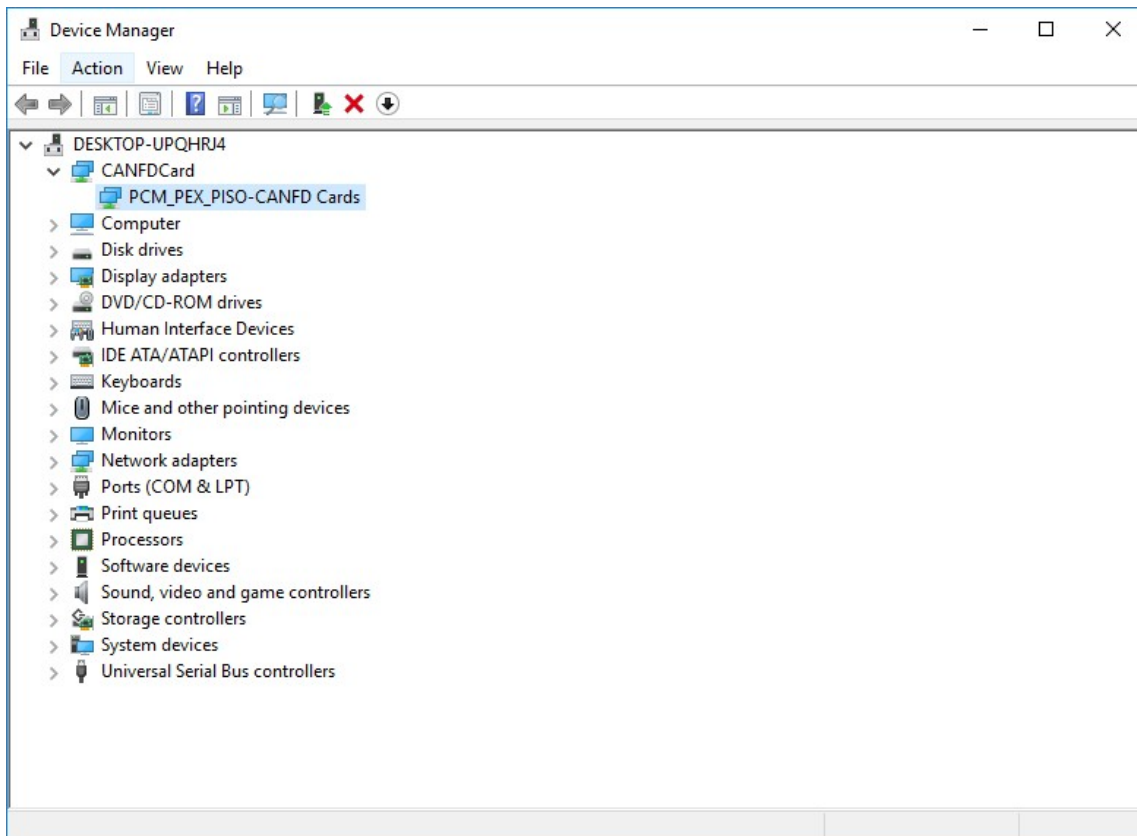
Step 4: Then Click the Browser...button to select the driver directory and click the **Next** button to start to install the driver.



Step 5: Once the installation has been completed, click the **Close** button to exit.

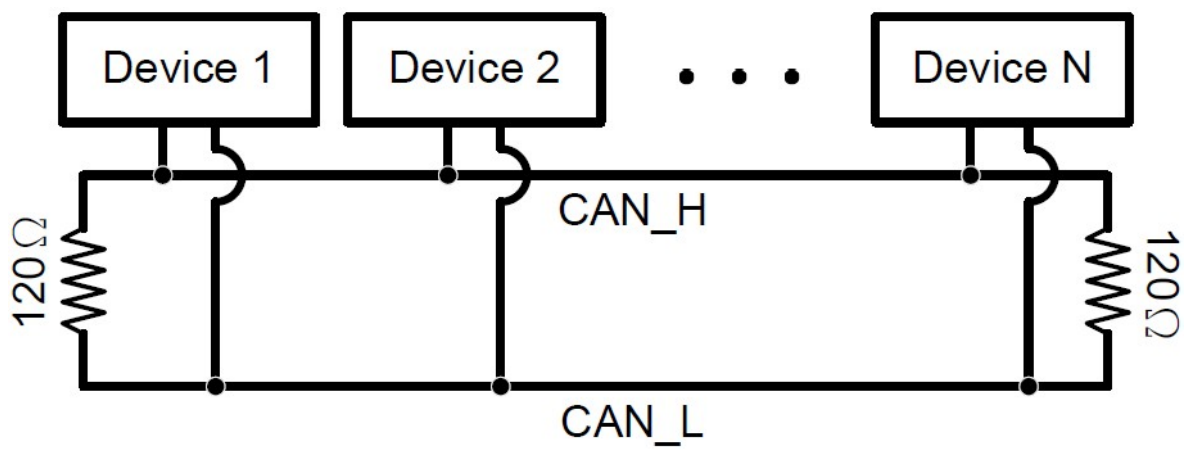


Step 6: After successfully to install the driver, you can see the **PCM_PEX_PISO-CANFD Cards** in **CANFDCard** item.



2.3. Wiring Connections

In order to minimize any reflection effects on the CAN bus line, it must be terminated at each end using a terminator resistor, as illustrated in the diagram below. The specifications provided in ISO 11898-2 state that each terminator resistor must be 120 Ω (or between 108 Ω and 132 Ω). The bus topology and the positions of these terminator resistors are shown below.



To minimize any voltage drop over long distances, the terminal resistance should be greater than the resistance value defined in the ISO 11898-2. The following table can be used as a good reference.

Bus Length (meter)	Bus Cable Parameters		Terminal Resistance (Ω)
	Length Related Resistance (mΩ/m)	Cross Section (Type)	
0~40	70	0.25(23AWG)~ 0.34mm ² (22AWG)	124 (0.1%)
40~300	<60	0.34(22AWG)~ 0.6mm ² (20AWG)	127 (0.1%)
300~600	<40	0.5~0.6mm ² (20AWG)	150~300
600~1K	<20	0.75~0.8mm ² (18AWG)	150~300

2.4. Software Utility

PISO-CANFD Utility is provided by ICP DAS to transmit / receive CAN/CAN FD messages for CAN Bus communication testing easily and quickly. In the meanwhile, it can also display the time-stamp of each received CAN/CAN FD messages for data analyzing conveniently.

Step 1: Get the PISO-CANFD Utility

The software is located at:

<https://www.icpdas.com/en/download/show.php?num=3199>

Step 2: Install .NET Framework 3.5 component

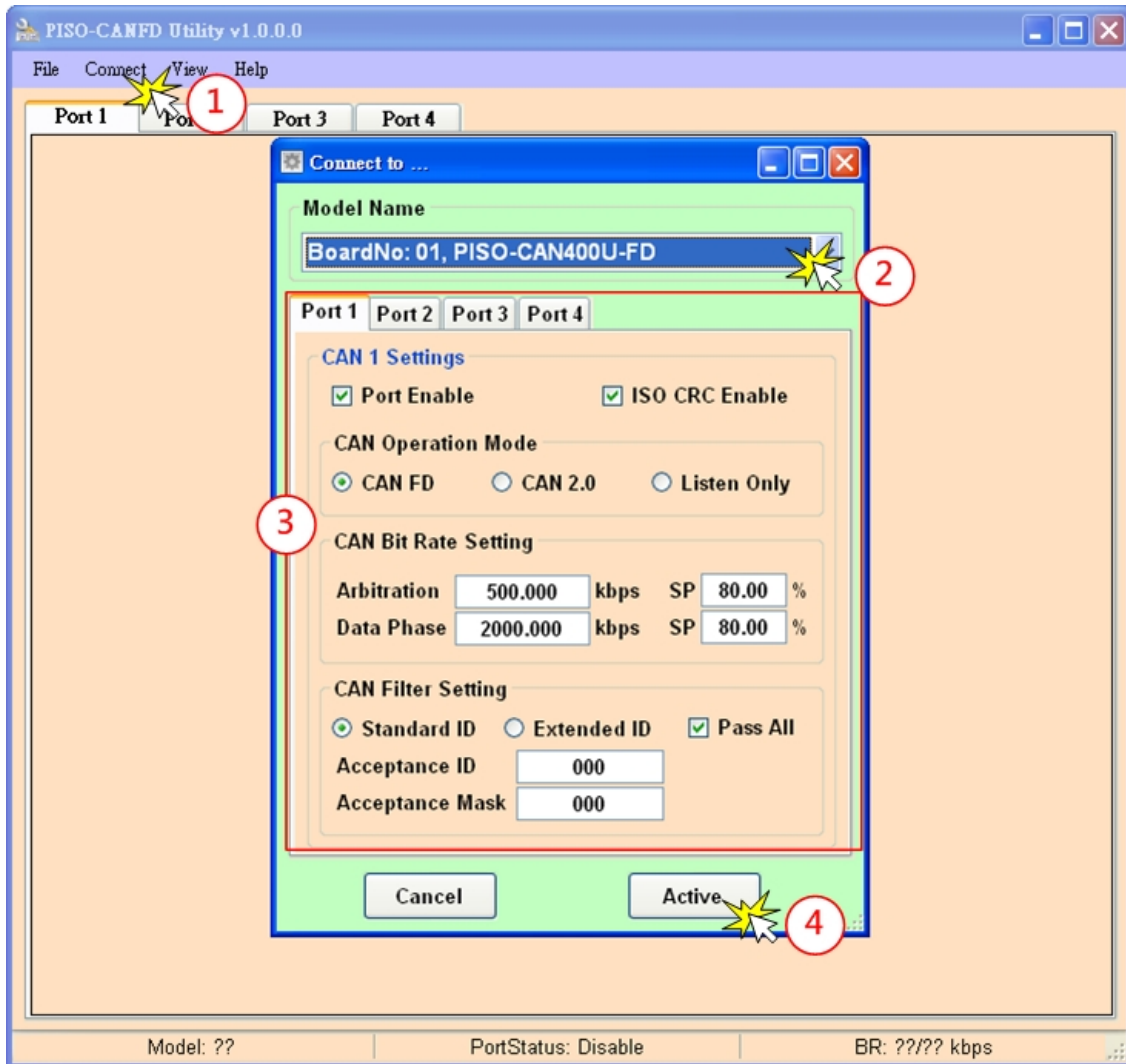
The PISO-CANFD Utility tool requires the .NET Framework 3.5 components. Before using the tool, user need to install .NET Framework 3.5 components from microsoft web site.

Step 3: Execute Utility tool

After installing the .Net Framework components, user can start to use the Utility tool to test the board.

2.4.1. Connect to the board

When executing the Utility, the tool will try to scan all the necessary PISO-CANFD series board (including PISO-CAN200U-FD, PISO-CAN400U-FD) and list all scanned boards name on “Model Name” location of the Utility “Connect” frame.



Before active the board, user can set the CAN port operation mode and CAN baudrate parameter of the board. Please refer to the following steps to configure the PSIO-CANFD series board.

- Step1: Click the “Connect to ...” item to open the “Connect” frame of Utility.
- Step2: Select the necessary PISO-CANFD series board.
- Step3: On the “CAN Settings” frame of each port, user can set the CAN Bus parameters. (The detail functions of these parameters are list below.)
- Step4: Press the “Active” button to start to use the CAN setting to send/receice CAN/CAN FD messages.

“Port Enable” : Enable or disable the selected CAN port.

“ISO CRC Enable” :

ISO CRC operation mode. If this parameter is checked, this port will use the CAN FD frame format as specified by the ISO11898-1. Otherwise, CAN FD frame format will follow according to Bosch CAN FD Specification V1.0.

[CAN Operation Mode]

“CAN FD” : Supports mixing of CAN FD and Classic CAN 2.0 frames. In this mode, the device will be on the CAN bus. It can transmit and receive messages in CAN FD mode; bit rate switching can be enabled and up to 64 data bytes can be transmitted and received.

“CAN 2.0” : Supports Classic CAN 2.0 frames. This is a the Classic CAN 2.0 mode. This port will not receive CAN FD frames. It might send error frames if CAN FD frames are detected on the bus.

“Listen Only” : Listen Only mode is a variant of Normal CAN FD Operation mode. If the Listen Only mode is activated, this port on the CAN bus is passive. It will receive messages, but it will not transmit any bits.

[CAN Bit Rate Setting]

“Arbitration” : CAN/CAN FD arbitration phase bit rate. Valid range: 10 kbps ~ 1000 kbps.

“Data Phase” : CAN FD data phase bit rate. Valid range: 100 kbps ~ 10 Mbps

“SP” : CAN/CAN FD arbitration/data phase bit rate sample point. Suggested range: 75.00 ~ 87.50 %

[CAN Filter Setting]

“Pass All” : Check this item will let this port to accept all (standard / extended, remote / data) frames

“Acceptance ID” : This field is used to determine which CAN IDs will be accepted by the CAN controller.

<i>ID Format</i>	<i>Acceptance ID (hexadecimal) Range</i>
Standard ID	000 to 7FF
Extended ID	00000000 to 1FFFFFFF

“Acceptance Mask” :

Specifies the “**Acceptance Mask**” that is used to determine which bit from the CAN ID will be checked by the CAN controller based on the value specified for the “**Acceptance ID**” parameter. If the bit of the “**Acceptance Mask**” parameter is set to 1, it means that the bit in the CAN ID in the same position needs to be checked, and the ID bit value needs to match the value specified for the bit of the “**Acceptance ID**” parameter in the same position.

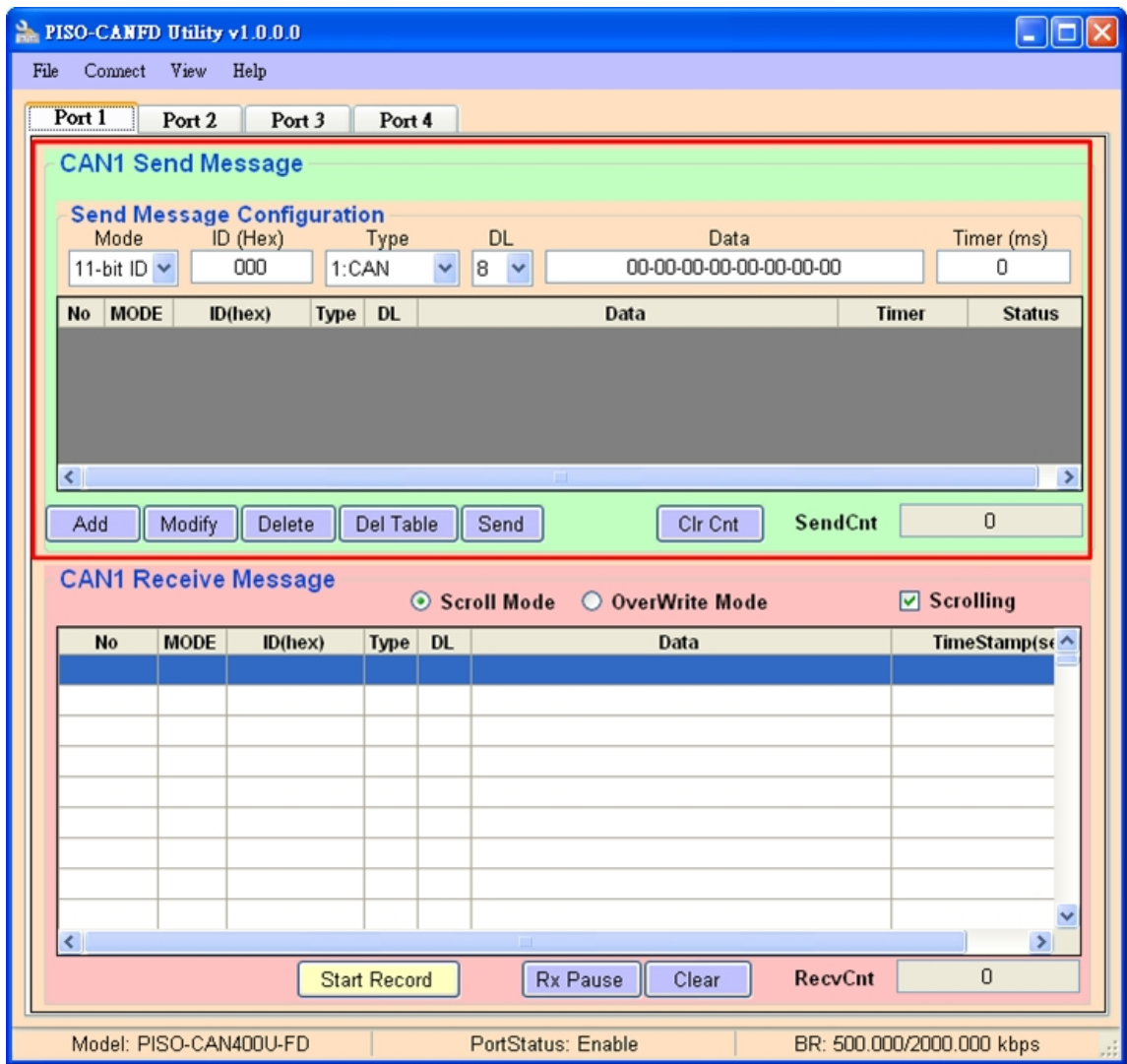
<i>ID Format</i>	<i>Acceptance Mask (hexadecimal) Range</i>
Standard ID	000 to 7FF
Extended ID	00000000 to 1FFFFFFF

Note that “**Acceptance ID**” and “**Acceptance Mask**” arguments should verify:
“**Acceptance ID**” & “**Acceptance Mask**” == “**Acceptance ID**”

And the “**Acceptance ID**” and “**Acceptance Mask**” arguments defines a filter that accepts frame whose identifier verifies:
identifier & “**Acceptance Mask**” == “**Acceptance ID**”

2.4.2. Send CAN/CAN FD messages

By using the Utility tool, user can send CAN/CAN FD messages to CAN Bus. After active the board, the screen for CAN Bus communication function will show up like below picture.



The above is the illustration for the “Communication” screen and it can be divided to two blocks in each CAN port function. One is “Send Message” block and the other is “Receive Message” block. Besides, “Port 1” to “Port 4” tab is used to switch CAN1 to CAN 4 “Communication” screen. Then user can send CAN/CAN FD message via “Send Message” block

CAN1 Send Message

Send Message Configuration

Mode: 11-bit ID | ID (Hex): 000 | Type: 1:CAN | DL: 8 | Data: 00-00-00-00-00-00-00-00 | Timer (ms): 0

No	MODE	ID(hex)	Type	DL	Data	Timer	Status

Buttons: Add, Modify, Delete, Del Table, Send, Clr Cnt, SendCnt (0)

[Send Message] block:

<1> “Send Message Configuration” frame :

It is used to edit the CAN message parameters and users can use “Add” button to add the CAN message to “CAN/CAN FD Message Send Area”.

- Mode : CAN 11-bit (standard) ID or 29-bit (extended) ID.
- ID : CAN ID field of CAN/CAN FD frame.
- Type : Frame type.
 - 0: Indicates that the message is a remote-transmit-request message
 - 1: Indicates that the message is a normal CAN data message
 - 2: Indicates that the message is a CAN FD message with no bit rate switch
 - 3: Indicates that the message is a CAN FD message with bit rate switch
- DL : CAN message data length.
For CAN frames, this field can be set 0 ~ 8 (means 0 ~ 8 bytes data length). For CAN FD frames, this field can be set 0 ~ 8, 12/16/20/24/32/48/64 (means 0 ~ 8, 12/16/20/24/32/48/64 bytes data length).
- Data : Data field of CAN messages. Each data must split with ‘-’.
- Timer(ms) : Transmission cycle of the CAN message. Unit: 1 millisecond.

<2> “Add” button :

It will add the CAN message from “Send Message Configuration” area to the last row in “CAN/CAN FD Message Send Area”.

<3> “Modify” button :

It will modify the CAN message parameter from “Send Message Configuration” area to the assigned blue row in “CAN/CAN FD Message Send Area”.

- <4> **“Delete”** button :
It will delete the CAN message of the assigned blue row in “CAN/CAN FD Message Send Area”.

- <5> **“Del Table”** button :
It will delete all the CAN messages in “CAN/CAN FD Message Send Area”.

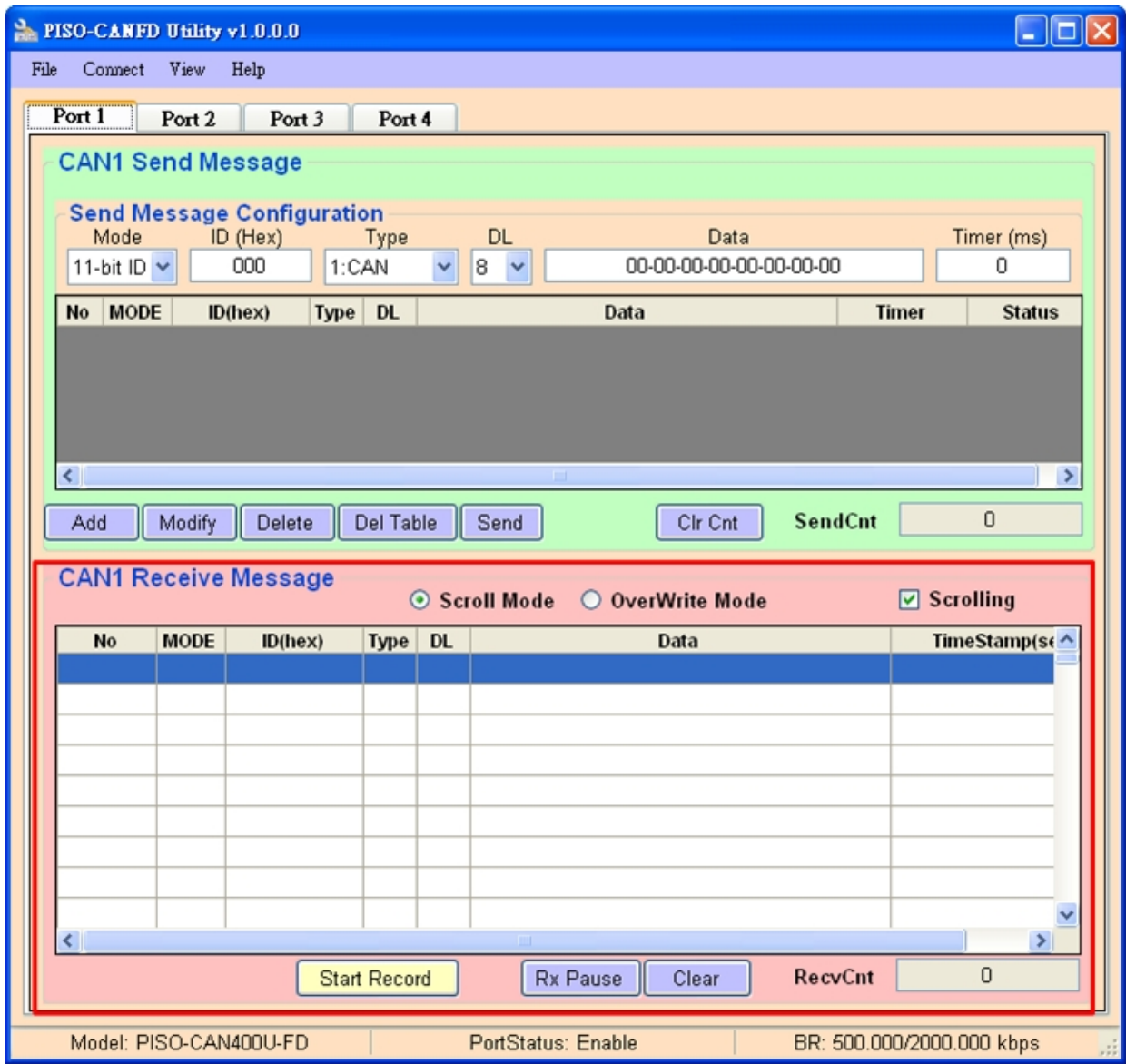
- <6> **“Send”** button :
It will send the CAN message of the assigned green row in “CAN/CAN FD Message Send Area”. If the value in the “Timer” field is zero, it will just send once. If not, it will send continuously by PC timer.

- <7> **“Clr Cnt”** button :
It will clear the “SendCnt” value to be zero in “CAN/CAN FD Message Send Area”.

- <8> **“SendCnt”** field :
Whenever the CAN message is sent out once, the “SendCnt” value will be added by 1 except “HWSend” function.

2.4.3. Receive CAN/CAN FD messages

By using the Utiltiy tool, user can review the received CAN meessages on the CAN Bus on the selected port. After active the board, the screen for CAN Bus communication function will show up like below picture.



<5> “**Scroll / OverWrite Mode**” option :

“**Scroll Mode**”:

The received CAN message data will be shown in “CAN Message Receive Area” by sequence.

“**Overwrite Mode**”:

If the MODE and ID value are all the same of the received CAN message data, then they will be placed in the same row of “CAN Message Receive Area”. The “Num” field will be the number of the same CAN message and the “CycleTime” field includes the period and the Max./Min. time interval of the received same CAN ID messages. The “CycleTime” field description is as below.

[1] 0.1543 (Sec) => CAN Message Period.

[2] 1.7578 (Sec) => The Maximum time interval of received CAN messages.

[3] 0.1543 (Sec) => The Minimum time interval of received CAN messages.

CAN2 Receive Message

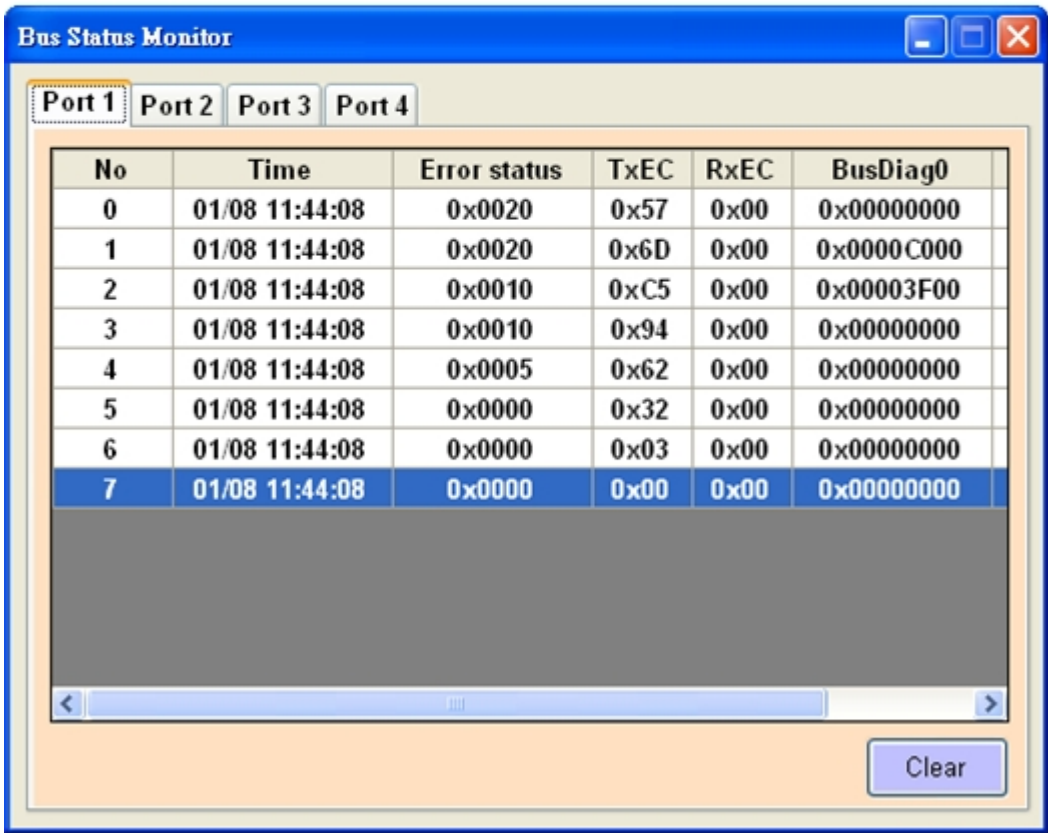
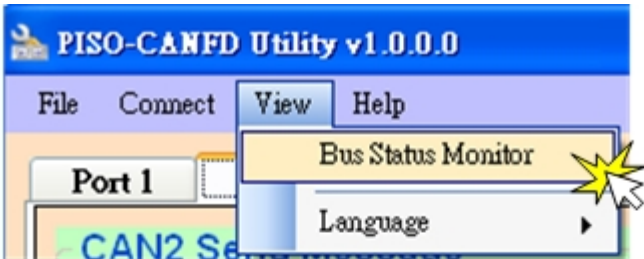
☐ Scroll Mode
☒ OverWrite Mode

☒ Scrolling

MODE	ID(hex)	Type	DL	Data	CycleTime(sec)
0	000	1	8	00-00-00-00-00-00-00-00	0.1543 (1.7578/0.1543)

2.4.4. Check CAN Bus Status

By using the Utiltiy tool, user can review the CAN Bus status of the selected port. After active the board, user can open the “CAN Bus Status Monitor” frame on the “View” => “Bus Status Monitor” item.



[Bus Status Monitor] block:

- <1> “No”: Number of CAN bus status.
- <2> “Time”: Event time of the CAN bus status.
- <3> “Error status”:

Bit	Symbol	Value	Description
0	EWARN		Transmitter or Receiver is in Error Warning State bit
		0	Transmitter or Receiver is not in Error Warning State
		1	Transmitter or Receiver is in Error Warning State

1	RXWARN		Receiver in Error Warning State bit
		0	Receiver not in Error Warning State (REC <= 95)
		1	Receiver in Error Warning State (128 > REC > 95)
2	TXWARN:		Transmitter in Error Warning State bit
		0	Transmitter not in Error Warning State (TEC <= 95)
		1	Transmitter in Error Warning State (128 > TEC > 95)
3	RXBP:		Receiver in Error Passive State bit
		0	Receiver not in Error Passive State (REC <= 127)
		1	Receiver in Error Passive State (REC > 127)
4	TXBP:		Transmitter in Error Passive State bit
		0	Transmitter in Error Passive State (TEC <= 127)
		1	Transmitter in Error Passive State (TEC > 127)
5	TXBO:		Transmitter in Bus Off State bit
		0	Transmitter not in Bus Off State
		1	Transmitter in Bus Off State (TEC > 255). In Configuration mode, TXBO is set, since the CAN module is not on the bus
15-6	-	-	reserved

<4> **“TxEc”** : Transmit error counter value of the CAN controller

<5> **“RxEc”** : Receiver error counter value of the CAN controller

<6> **“BusDiag0”** :

BusDiag0 contains separate error counters for receive/transmit and for nominal/data bit rates. They are simply incremented by one on every error.

Bit	Symbol	Description
7-0	NRERRCNT	Nominal Bit Rate Receive Error Counter bits
15-8	NTERRCNT	Nominal Bit Rate Transmit Error Counter bits
23-16	DRERRCNT:	Data Bit Rate Receive Error Counter bits
31-24	DTERRCNT:	Data Bit Rate Transmit Error Counter bits

<7> **“BusDiag1”** :

BusDiag1 keeps track of the kind of error that occurred since the last clearing of the register. The register also contains the error-free message counter. The flags and the counter are cleared after reading.

Bit	Symbol	Description
15-0	EFMSGCNT	Error Free Message Counter bits
16	NBIT0ERR:	During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a

		dominant level (data or identifier bit logical value '0'), but the monitored bus value was recessive.
17	NBIT1ERR::	During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value '1'), but the monitored bus value was dominant.
18	NACKERR::	Transmitted message was not acknowledged
19	NFORMERR:	A fixed format part of a received frame has the wrong format
20	NSTUFERR:	More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed
21	NCRCERR:	The CRC check sum of a received message was incorrect. The CRC of an incoming message does not match with the CRC calculated from the received data.
22	-	Reserved, read as '0'
23	TXBOERR:	Device went to bus-off (and auto-recovered).
24	DBIT0ERR:	Same as for nominal bit rate (see above).
25	DBIT1ERR:	Same as for nominal bit rate (see above).
26	-	Reserved, read as '0'
27	DFORMERR:	Same as for nominal bit rate (see above).
28	DSTUFERR:	Same as for nominal bit rate (see above).
29	DCRCERR:	Same as for nominal bit rate (see above).
30	ESI:	ESI flag of a received CAN FD message was set.
31	DLCMM:	transmission or reception DLC mismatch bit

3. Windows API Function Reference

This chapter describes the “pisocanfd” library APIs, including the System Information API, the CAN Bus API and an overview of the error codes, which can be helpful when developing custom applications. The library and demos can be downloaded from the ICP DAS web site.

The demos and library are located at:

<https://www.icpdas.com/en/download/show.php?num=3196>

3.1. API Library Overview

All the functions provided by pisocanfd library can be separated into two groups, “System Information API” and “CAN Bus API”.

[System Information API]

These functions are used to active and close the valid and necessary CAN board, and can be used to get the board system information like dll version, board id, board hardware version ... etc.

[CAN Bus API]

These functions are used to initialize, reset CAN hardware and configure CAN Bus bitrate, filter and operation mode parameters. Besides, user can use these API to send/receive CAN messages and diagnostic CAN bus status.

3.2. API Library Function Table

All the functions provided in the pisocanfd API library are listed in the following table.

System Information API	
Function	Description
CANFD_GetDllVersion	Used to retrieve the version number for the function library file currently installed on the PISO-CANFD series board
CANFD_GetBoardInf	Used to retrieve PCI information related to a selected PISO-CANFD series board, including the vendor ID, device ID, sub-vendor ID, sub-device ID, sub-auxiliary ID, logical interrupt number, board ID and board switch number.
CANFD_TotalBoard	Used to retrieve the total board number of PISO-CANFD series boards installed in the Host PC
CANFD_GetCardBoardSwitchNo	Used to retrieve the switch (SW1) number for the selected PISO-CANFD series board
CANFD_GetCardBoardID	Used to retrieve the card ID for the selected PISO-CANFD series board
CANFD_GetCardPortNum	Used to retrieve the total CAN port number of the selected PISO-CANFD series board
CANFD_GetCardFPGAFWVer	Used to get the FPGA firmware version of the selected PISO-CANFD series board
CANFD_ActiveBoard	Used to activate the selected PISO-CANFD series board
CANFD_CloseBoard	Used to deactivate the selected PISO-CANFD series board
CANFD_BoardIsActive	Used to check whether or not the selected PISO-CANFD series board is active

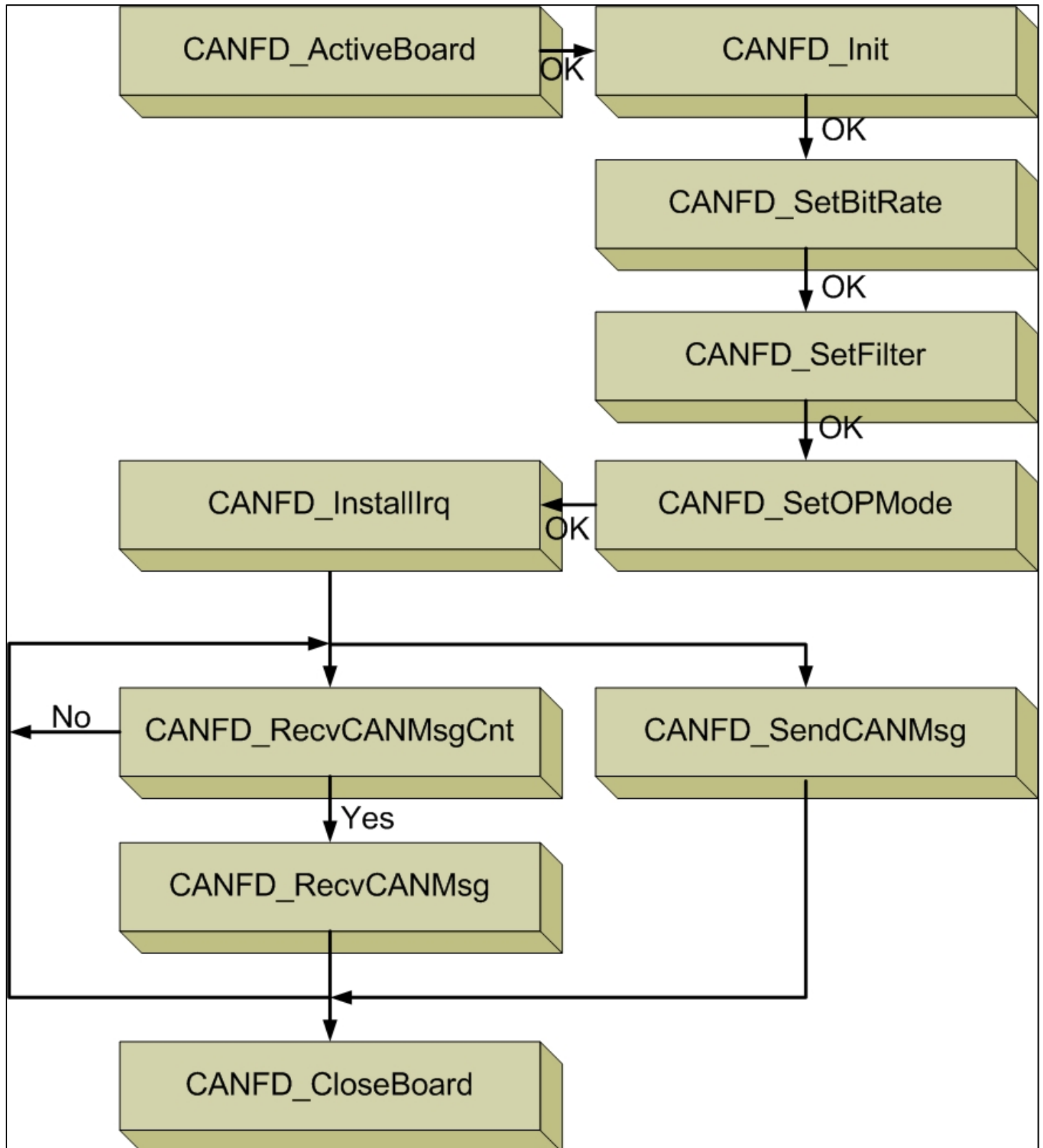
CAN Bus API	
Function	Description
CAN Reset, Initialize function	
CANFD_Reset	Used to reset the CAN controller on the selected PISO-CANFD series board
CANFD_Init	Used to initialize the CAN port function on the selected PISO-CANFD series board
CAN bit rate configuration functions	
CANFD_SetBitRate	Used to set the bit rate configuration for the CAN port on the selected PISO-CANFD series board
CANFD_SetBitRateWithSP	Used to set the bit rate and baudrate sample point configuration for the CAN port on the selected PISO-CANFD series board

CANFD_GetBitRate	Used to retrieve the current bit rate configuration of the CAN port on the selected PISO-CANFD series board
CANFD_GetBitRateWithSP	Used to retrieve the current bit rate and baudrate sample point configuration of the CAN port on the selected PISO-CANFD series board
CAN filter configuration functions	
CANFD_SetFilterAllPass	Used to set the CAN message filter of the CAN port on the selected PISO-CANFD series board to accept all messages.
CANFD_SetFilterFormat	Used to set the CAN message filter of the CAN port on the selected PISO-CANFD series board via standard and extended Identifier.
CANFD_SetFilter	Used to set the CAN message filter of the CAN port on the selected PISO-CANFD series board via Identifier acceptance and mask bits
CAN operation mode functions	
CANFD_SetOPMode	Used to set the operation mode (configuration, listen, CAN 2.0, CAN FD mode) of the CAN port on the selected PISO-CANFD series board
CANFD_GetOPMode	Used to retrieve the operation mode of the CAN port on the selected PISO-CANFD series board
CANFD_SetISOCRCEn	Used to enable ISO CRC in CAN FD frames of the CAN port on the selected PISO-CANFD series board
CANFD_GetISOCRCEn	Used to get enable state of ISO CRC in CAN FD frames of the CAN port on the selected PISO-CANFD series board
CAN receive interrupt functions	
CANFD_InstallIrq	Used to enable the function of getting CAN messages and save to receive buffer via interrupt method on driver.
CANFD_RemoveIrq	Used to disable the function of getting CAN messages via interrupt method on driver.
CANFD_GetIrqStatus	Used to get the state of the function of getting CAN messages via interrupt method.
CAN send/receive data functions	
CANFD_SendCANMsg	Used to send a CAN message from the CAN port on the selected PISO-CANFD series board
CANFD_RecvCANMsg	Used to receive a CAN message from the CAN port on the selected PISO-CANFD series board
CANFD_RecvCANMsgCnt	Used to get the received CAN message counter from the CAN port on the selected PISO-CANFD series board
CAN FIFO functions	
CANFD_SetFIFOStatus	Used to clear the receive and transmit buffer overflow state of the CAN port on the selected PISO-CANFD series board
CANFD_GetFIFOStatus	Used to get the receive and transmit buffer overflow state of the CAN

	port on the selected PISO-CANFD series board
CAN bus status functions	
CANFD_GetCANStatus	Used to retrieve the status of the CAN port on the selected PISO-CANFD series board
CANFD_GetBUSDiagnostic	Used to retrieve the bus diagnostic status of the CAN port on the selected PISO-CANFD series board

3.3. API Library Flow Diagram

The following is the basic control flow chart of user's CAN Bus program development by using pisocanfd API library shown in following picture.



3.4. System Information API

The following is an overview of the System Information API functions provided on the PISO-CANFD series board. A detailed description of each API is presented in subsequent sections.

3.1.1. CANFD_GetDllVersion

Description

This function is used to retrieve the version number of the pisocanfd.dll driver currently installed for the PISO-CANFD series board.

Syntax

```
C#
.....
UInt16 CANFD_GetDllVersion(out UInt16 DLLVer);
```

Parameters

DLLVer

[out] The version information for the pisocanfd.dll driver currently installed for the PISO-CANFD series board in hexadecimal format. For example, the value of “0x100” indicates that the driver version is “1.00”.

Return Value

If the function succeeds, the return value will be 0.
If the function fails, refer to Section 3.6 Error Code Definitions.

3.1.2. CANFD_GetBoardInf

Description

This function is used to retrieve PCI information of related to a specified PISO-CANFD series board, including the vendor ID, device ID, sub-vendor ID, sub-device ID, sub-auxiliary ID, logical interrupt number, board ID and board switch number.

Syntax

```
C#
.....
Int16 CANFD_GetBoardInf(
    Byte BoardNo,
    out UInt16 wVID,
    out UInt16 wDID,
    out UInt16 wSVID,
    out UInt16 wSDID,
    out UInt16 wSAuxID,
    out UInt16 wlrqNo,
    out Byte BoardID,
    out Byte BoardSwitchNo
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

wVID

[out] Indicates the address of a variable used to receive the vendor ID.

wDID

[out] Indicates the address of a variable used to receive the device ID.

wSVID

[out] Indicates the address of a variable used to receive the sub-vendor ID.

wSDID

[out] Indicates the address of a variable used to receive the sub-device ID.

wSAuxID

[out] Indicates the address of a variable used to receive the sub-auxiliary ID.

wIrqNo

[out] Indicates the address of a variable used to receive the logical interrupt number.

BoardID

[out] Indicates the board id of the selected PISO-CANFD series board.

Board ID	Model Name
1	PISO-CAN200U-FD
2	PISO-CAN400U-FD
3	PEX-CAN200i-FD
4	PCM-CAN200-FD

BoardSwitchNo

[out] Indicates the address of a variable used to receive the board switch number of the selected PISO-CANFD series board.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.1.3. CANFD_TotalBoard

Description

This function is used to retrieve the total board number of PISO-CANFD series boards currently installed in the Host PC.

Syntax

```
C#  
.....  
Int16 CANFD_TotalBoard (  
    out Byte BoardNo  
);
```

Parameters

BoardNo

[out] Indicates the address of a variable used to receive the total number of PISO-CANFD series boards that were scanned.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.1.4. CANFD_GetCardBoardSwitchNo

Description

This function is used to retrieve the current configuration of the DIP switch (SW1) on the specified PISO-CANFD series board.

Syntax

```
C#  
.....  
Int16 CANFD_GetCardBoardSwitchNo(  
    Byte BoardNo,  
    out Byte BoardSwitchNo  
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

BoardSwitchNo

[out] Indicates the address of a variable used to retrieve the current configuration of the DIP switch (SW1) on the PISO-CANFD series board.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.1.5. CANFD_GetCardBoardID

Description

This function is used to retrieve the board id of the selected PISO-CANFD series board.

Syntax

```
C#
.....
Int16 CANFD_GetCardBoardID(
    Byte BoardNo,
    out Byte BoardID
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

BoardID

[out] Indicates the address of a variable used to retrieve the board idof the selected PISO-CANFD series board.

Board ID	Model Name
1	PISO-CAN200U-FD
2	PISO-CAN400U-FD
3	PEX-CAN200i-FD
4	PCM-CAN200-FD

Return Value

If the function succeeds, the return value will be 0.
If the function fails, refer to Section 3.6 Error Code Definitions.

3.1.6. CANFD_GetCardPortNum

Description

This function is used to retrieve the number of the CAN port on a specified PISO-CANFD series board.

Syntax

```
C#  
.....  
Int16 CANFD_GetCardPortNum(  
    Byte BoardNo,  
    out Byte PortNum  
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

PortNum

[out] Indicates the address of a variable used to retrieve the number of the CAN port on the PISO-CANFD series board.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.1.7. CANFD_GetCardFPGAFWVer

Description

This function is used to retrieve the FPGA firmware version on a specified PISO-CANFD series board.

Syntax

```
C#  
.....  
Int16 CANFD_GetCardFPGAFWVer(  
    Byte BoardNo,  
    out Byte FPGAFWVer  
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

FPGAFWVer

[out] Indicates the address of a variable used to retrieve the FPGA firmware version of the selected PISO-CANFD series board.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.1.8. CANFD_ActiveBoard

Description

This function is used to activate a specified PISO-CANFD series board. Note that this function MUST be called before using any other API functions.

Syntax

```
C#
.....
Int16 CANFD_ActiveBoard(
    Byte BoardNo
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.1.9. CANFD_CloseBoard

Description

This function is used to deactivate a specified PISO-CANFD series board. This function MUST always be called at the end of a program in order for the system to release resources before exiting the program.

Syntax

```
C#
.....
Int16 CANFD_CloseBoard(
    Byte BoardNo
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.1.10. CANFD_BoardIsActive

Description

This function is used to check whether or not a specified PISO-CANFD series board is active.

Syntax

```
C#
.....
Int16 CANFD_BoardIsActive(
    Byte BoardNo,
    out Byte IsActive
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

IsActive

[out] Indicates the address of a variable used to retrieve the status of the selected PISO-CANFD series board, where:

0: The selected board is not active

1: The selected board has been activated

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.5. CAN Bus API

The following is an overview of the CAN Bus API functions provided on the PISO-CANFD series board. A detailed description of each function is presented in subsequent sections.

3.2.1. CANFD_Reset

Description

This function is used to reset the CAN chip for the CAN port on a specified PISO-CANFD series board. After resetting, this CAN port will be in configuration mode.

Syntax

```
C#
.....
Int16 CANFD_Reset (
    Byte BoardNo,
    Byte Port
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1.
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.2. CANFD_Init

Description

This function is used to initialize the CAN chip of the CAN port on a specified PISO-CANFD series board. After initializing, this CAN port will be in configuration mode.

Syntax

```
C#
.....
Int16 CANFD_Init (
    Byte BoardNo,
    Byte Port
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1.
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.3. CANFD_SetBitRate

Description

This function is used to configure the CAN/CAN FD normal/data bit rate of the CAN port on a specified PISO-CANFD series board. After setting, this CAN port will be in configuration mode.

Syntax

```
C#
.....
Int16 CANFD_SetBitRate(
    Byte BoardNo,
    Byte Port,
    UInt32 NormalBitRate,
    UInt32 DataBitRate
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

NormalBitRate

[in] The bit rate configured for the CAN and CAN FD normal arbitration phase in the assigned CAN port of the PISO-CANFD series card.

Unit: bps (bit per second).

Valid Range: 10000 ~ 1000000 (10 kbps ~ 1000 kbps).

DataBitRate

[in] The bit rate configured for the CAN FD data phase in the assigned CAN port

of the PISO-CANFD series card.

Unit: bps (bit per second).

Valid Range: 100000 ~ 100000000 (100 kbps ~ 10000 kbps).

Remark:

The bit rate configured for CAN FD data phase (**DataBitRate**) must be higher or equal to the bit rate configured for the normal arbitration phase (**NormalBitRate**).

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.5. CANFD_SetBitRateWithSP

Description

This function is used to configure the CAN/CAN FD normal/data bit rate and sample point of the CAN port on a specified PISO-CANFD series board. After setting, this CAN port will be in configuration mode.

Syntax

```
C#
.....
Int16 CANFD_SetBitRateWithSP(
    Byte BoardNo,
    Byte Port,
    UInt32 NormalBitRate,
    UInt32 DataBitRate,
    UInt16 NormalSP,
    UInt16 DataSP
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

NormalBitRate

[in] The bit rate configured for the CAN and CAN FD normal arbitration phase in the assigned CAN port of the PISO-CANFD series card.

Unit: bps (bit per second).

Valid Range: 10000 ~ 1000000 (10 kbps ~ 1000 kbps).

DataBitRate

[in] The bit rate configured for the CAN FD data phase in the assigned CAN port of the PISO-CANFD series card.

Unit: bps (bit per second).

Valid Range: 100000 ~ 10000000 (100 kbps ~ 10000 kbps).

Remark:

The bit rate configured for CAN FD data phase (**DataBitRate**) must be higher or equal to the bit rate configured for the normal arbitration phase (**NormalBitRate**).

NormalSP

[in] Specifies the normal arbitration phase bit rate sample point as a percentage. The valid range is from 0.00% to 100.00% in intervals of 0.01%. Value 8000 means 80.00%.

DataSP

[in] Specifies the normal data phase bit rate sample point as a percentage. The valid range is from 0.00% to 100.00% in intervals of 0.01%. Value 8000 means 80.00%.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.6. CANFD_GetBitRate

Description

This function is used to retrieve the CAN/CAN FD baud rate configuration of the CAN port on a specified PISO-CANFD series board.

Syntax

```
C#
.....
Int16 CANFD_GetBitRate(
    Byte BoardNo,
    Byte Port,
    out UInt32 NormalBitRate,
    out UInt32 DataBitRate
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

NormalBitRate

[out] Indicates the address of a variable used to retrieve the bit rate configured for the CAN and CAN FD normal arbitration phase in the assigned CAN port of the PISO-CANFD series card.

Unit: bps (bit per second).

DataBitRate

[out] Indicates the address of a variable used to retrieve the bit rate configured for the CAN FD data phase in the assigned CAN port of the PISO-CANFD

series card.

Unit: bps (bit per second).

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.7. CANFD_GetBitRateWithSP

Description

This function is used to retrieve the CAN/CAN FD baud rate configuration and bit sample point of the CAN port on a specified PISO-CANFD series board.

Syntax

```
C#
.....
Int16 CANFD_GetBitRateWithSP(
    Byte BoardNo,
    Byte Port,
    out UInt32 NormalBitRate,
    out UInt32 DataBitRate,
    out UInt16 NormalSP,
    out UInt16 DataSP,
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

NormalBitRate

[out] Indicates the address of a variable used to retrieve the bit rate configured for the CAN and CAN FD normal arbitration phase in the assigned CAN port of the PISO-CANFD series card.

Unit: bps (bit per second).

DataBitRate

[out] Indicates the address of a variable used to retrieve the bit rate configured for the CAN FD data phase in the assigned CAN port of the PISO-CANFD series card.

Unit: bps (bit per second).

NormalSP

[out] Indicates the address of a variable used to retrieve the normal arbitration phase bit sample point that is currently configured for the CAN Bus.

Unit: 0.01%. Value 8000 means 80.00%

DataSP

[out] Indicates the address of a variable used to retrieve the data phase bit sample point that is currently configured for the CAN Bus.

Unit: 0.01%. Value 8000 means 80.00%

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.8. CANFD_SetFilterAllPass

Description

This function is used to configure the CAN port on a specified PISO-CANFD series board to accept all (standard / extended, remote / data) frames.

Syntax

```
C#
.....
Int16 CANFD_SetFilterAllPass(
    Byte BoardNo,
    Byte Port
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.9. CANFD_SetFilterFormat

Description

This function is used to configure the CAN port on a specified PISO-CANFD series board to accept standard or extended frames.

Syntax

```
C#
.....
Int16 CANFD_SetFilterFormat(
    Byte BoardNo,
    Byte Port,
    Byte inFormat
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

inFormat

[in] Specifies the CAN port to accept standard or extended remote frame and data frames.

- 0: standard frame
- 1: extended frame

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.10. CANFD_SetFilter

Description

This function is used to configure the CAN port on a specified PISO-CANFD series board to accept standard or extended frames depended on acceptance and mask parameter setting.

Syntax

```
C#
.....
Int16 CANFD_SetFilter (
    Byte BoardNo,
    Byte Port,
    Byte inFormat,
    UInt32 inAcceptance,
    UInt32 inMask
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

inFormat

[in] Specifies the CAN port to accept standard or extended remote frame and data frames.

- 0: standard frame (11-bit CAN ID)
- 1: extended frame (29-bit CAN ID)

inAcceptance

[in] Specifies the “inAcceptance” that is used to determine which CAN IDs will be accepted by the CAN controller.

<i>inFormat</i>	<i>inAcceptance (hexadecimal) Range</i>
standard frame (0)	000 to 7FF
extended frame (1)	00000000 to 1FFFFFFF

Note that inMask and inAcceptance arguments should verify:
 $\text{inAcceptance} \& \text{inMask} == \text{inAcceptance}$

inMask

[in] Specifies the “inMask” that is used to determine which bit from the CAN ID will be checked by the CAN controller based on the value specified for the “inAcceptance” parameter. If the bit of the “inMask” parameter is set to 1, it means that the bit in the CAN ID in the same position needs to be checked, and the ID bit value needs to match the value specified for the bit of the “inAcceptance” parameter in the same position.

<i>inFormat</i>	<i>inMask (hexadecimal) Range</i>
standard frame (0)	000 to 7FF
extended frame (1)	00000000 to 1FFFFFFF

Note that inMask and inAcceptance arguments should verify:
 $\text{inAcceptance} \& \text{inMask} == \text{inAcceptance}$

The inMask and inAcceptance arguments defines a filter that accepts frame whose identifier verifies:
 $\text{“identifier} \& \text{inMask} == \text{inAcceptance”}$

Return Value

If the function succeeds, the return value will be 0.
If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.11. CANFD_SetOPMode

Description

This function is used to configure the operation mode of the CAN port on a specified PISO-CANFD series board to CAN FD, CAN 2.0 and listen only mode.

Syntax

```
C#
.....
Int16 CANFD_SetOPMode(
    Byte BoardNo,
    Byte Port,
    Byte OPMODE
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

OPMode

[in] Specifies the CAN port to CAN FD, CAN 2.0 and listen only mode.

OPMode	Operation Mode	Description
0	CAN FD	Supports mixing of CAN FD and Classic CAN 2.0 frames. In this mode, the device will be on the CAN bus. It can transmit and receive messages in CAN FD mode; bit rate switching can be enabled and up to 64 data bytes can be transmitted and received.

1	CAN 2.0	Supports Classic CAN 2.0 frames. This is a the Classic CAN 2.0 mode. This port will not receive CAN FD frames. It might send error frames if CAN FD frames are detected on the bus.
2	Listen Only	Listen Only mode is a variant of Normal CAN FD Operation mode. If the Listen Only mode is activated, this port on the CAN bus is passive. It will receive messages, but it will not transmit any bits.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.12. CANFD_GetOPMode

Description

This function is used to get the operation mode of the CAN port on a specified PISO-CANFD series board

Syntax

```
C#
.....
Int16 CANFD_GetOPMode(
    Byte BoardNo,
    Byte Port,
    out Byte OPMode
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

OPMode

[out] Indicates the address of a variable used to retrieve the CAN port operation mode.

OPMode	Operation Mode	Description
0	CAN FD	Supports mixing of CAN FD and Classic CAN 2.0 frames. In this mode, the device will be on the CAN bus. It can transmit and receive messages in CAN FD mode; bit rate switching can be enabled and up to 64 data bytes can be transmitted and

		received.
1	CAN 2.0	Supports Classic CAN 2.0 frames. This is a the Classic CAN 2.0 mode. This port will not receive CAN FD frames. It might send error frames if CAN FD frames are detected on the bus.
2	Listen Only	Listen Only mode is a variant of Normal CAN FD Operation mode. If the Listen Only mode is activated, the CAN port on the CAN bus is passive. It will receive messages, but it will not transmit any bits.
3	Configuration	After reset, the CAN FD Controller module is in Configuration mode. The CAN FD Controller module has to be initialized before setting.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.13. CANFD_SetISOCRCEn

Description

This function is used to enable/disable the ISO/Non-ISO CRC in CAN FD Frames bit of the CAN port on a specified PISO-CANFD series board

Syntax

```
C#
.....
Int16 CANFD_SetISOCRCEn(
    Byte BoardNo,
    Byte Port,
    Byte ISOCRCEn
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

ISOCRCEn

[in] Specifies the CAN port to enable or disable ISO CRC in CAN FD Frames bit.

ISOCRCEn	Description
0	Do NOT include Stuff Bit Count in CRC Field and use CRC Initialization Vector with all zeros
1	Include Stuff Bit Count in CRC Field and use Non-Zero CRC Initialization Vector according to ISO 11898-1:2015

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.14. CANFD_GetISOCRCEn

Description

This function is used to get the ISO/Non-ISO CRC setting of the CAN port on a specified PISO-CANFD series board

Syntax

```
C#
.....
Int16 CANFD_GetISOCRCEn(
    Byte BoardNo,
    Byte Port,
    out Byte ISOCRCEn
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3.
- 4: CAN port 4.

ISOCRCEn

[out] Indicates the address of a variable used to retrieve the CAN port is enable or disable ISO CRC in CAN FD Frames bit or not.

ISOCRCEn	Description
0	Do NOT include Stuff Bit Count in CRC Field and use CRC Initialization Vector with all zeros
1	Include Stuff Bit Count in CRC Field and use Non-Zero CRC Initialization Vector according to ISO 11898-1:2015

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.15. CANFD_SendCANMsg

Description

This function is used to transmit a CAN message from the CAN controller of a specified PISO-CANFD CAN port to the CAN network.

Syntax

```
C#
.....
Int16 CANFD_SendCANMsg (
    Byte BoardNo,
    Byte Port,
    Byte Mode,
    UInt32 Id,
    Byte Type,
    Byte Dlc,
    Byte[] Data,
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3
- 4: CAN port 4.

Mode

[in] Indicates the address of a variable used to retrieve the CAN standard/extended message ID.

0: Denotes the CAN standard ID, which uses a 11-bit CAN message ID.

1: Denotes the CAN extended ID, which uses a 29-bit CAN message ID.

Id

[in] CAN message ID parameter.

standard CAN message (11-bit CAN ID) → 0x000 ~ 0x7FF

extended CAN message (29-bit CAN ID) → 0x00000000 ~ 0x1FFFFFFF

Type

[in] The type of the message, where:

0: Indicates that the message is a remote-transmit-request message

1: Indicates that the message is a normal CAN data message

2: Indicates that the message is a CAN FD message with no bit rate switch

3: Indicates that the message is a CAN FD message with bit rate switch

Note that as this parameter will be set to 0, the *Data* parameter will be useless.

Dlc

[in] CAN/CAN FD frame data length code parameter.

Normal CAN frame: Valid range: 0x0 ~ 0x8

CAN FD frame: Valid range: 0x0 ~ 0xF

<i>Dlc</i> (Hexadecimal)	<i>Frame data length</i> (Decimal)	<i>Dlc</i> (Hexadecimal)	<i>Frame data length</i> (Decimal)
0x0	0	0x8	8
0x1	1	0x9	12
0x2	2	0xA	16
0x3	3	0xB	20
0x4	4	0xC	24
0x5	5	0xD	32
0x6	6	0xE	48
0x7	7	0xF	64

Data

[in] This point to an user defined 64 bytes array buffer for saving CAN/CAN FD message data parameter.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.16. CANFD_RecvCANMsg

Description

This function is used to retrieve a CAN message from the buffer of driver or CAN controller for a specified PISO-CANFD CAN port.

Syntax

```
C#
.....
Int16 CANFD_RecvCANMsg (
    Byte BoardNo,
    Byte Port,
    out Byte Mode,
    out UInt32 Id,
    out Byte Type,
    out Byte Dlc,
    Byte[] Data,
    out UInt32 H_MsgTimeStamps,
    out UInt32 L_MsgTimeStamps
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3
- 4: CAN port 4.

Mode

[out] Indicates the address of a variable used to retrieve the CAN standard/extended message ID.

- 0: Denotes the CAN standard ID, which uses a 11-bit CAN message ID.
 1: Denotes the CAN extended ID, which uses a 29-bit CAN message ID.

Id

[out] CAN message ID parameter.

standard CAN message (11-bit CAN ID) → 0x000 ~ 0x7FF

extended CAN message (29-bit CAN ID) → 0x00000000 ~ 0x1FFFFFFF

Type

[out] Indicates the address of a variable used to retrieve the type of the message, where:

0: Indicates that the message is a remote-transmit-request message

1: Indicates that the message is a normal CAN data message

2: Indicates that the message is a CAN FD message with no bit rate switch

3: Indicates that the message is a CAN FD message with bit rate switch

Note that as this parameter will be set to 0, the *Data* parameter will be useless.

Dlc

[out] CAN/CAN FD frame data length code parameter.

Normal CAN frame: Valid range: 0x0 ~ 0x8

CAN FD frame: Valid range: 0x0 ~ 0xF

<i>Dlc</i> (Hexadecimal)	<i>Frame data length</i> (Decimal)	<i>Dlc</i> (Hexadecimal)	<i>Frame data length</i> (Decimal)
0x0	0	0x8	8
0x1	1	0x9	12
0x2	2	0xA	16
0x3	3	0xB	20
0x4	4	0xC	24
0x5	5	0xD	32
0x6	6	0xE	48
0x7	7	0xF	64

Data

[out] This point to an user defined 64 bytes array buffer for saving CAN/CAN FD message data parameter.

H_MsgTimeStamps

[out] Indicates the address of a variable used to retrieve the upper timestamp of the received CAN message in increments interval of 4,294,967,295

micro-seconds.

L_MsgTimeStamps

[out] Indicates the address of a variable used to retrieve the lower timestamp of the received CAN message in increments interval of 100 nano-seconds. The maximum value is 4,294,967,295 micro-seconds

**NOTE: The total timestamp of the received CAN message equals to
“(UInt64) ((H_MsgTimeStamps << 32) + L_MsgTimeStamps)”,
unit: 0.1 micro-second.**

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.17. CANFD_RecvCANMsgCnt

Description

This function is used to retrieve the count of the received CAN message for a specified PISO-CANFD CAN port. When user using IRQ method (call CANFD_InstallIrq) to receive CAN message, the value of “MsgCnt” will be the sum of message count in receive buffer of driver and hardware buffer of CAN controller. If user not install IRQ, this value will be the message count in hardware buffer of CAN controller.

Syntax

```
C#
.....
Int16 CANFD_RecvCANMsgCnt (
    Byte BoardNo,
    Byte Port,
    out UInt16 MsgCnt
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3
- 4: CAN port 4.

MsgCnt

[out] Indicates the count of the received CAN messages.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.18. CANFD_SetFIFOStatus

Description

This function is used to clear the receive/ transmit data buffer of the CAN port on a specified PISO-CANFD series board.

Syntax

```
C#
.....
Int16 CANFD_SetFIFOStatus (
    Byte BoardNo,
    Byte Port,
    Byte FIFOState
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2
- 3: CAN port 3
- 4: CAN port 4.

FIFOState

[in] Specifies the receive/ transmit data buffer that needs to clear.

Bit	Symbol	Value	Description
0	RXBUFF		receive buffer
		0	No need to clear the receive buffer
		1	clear the receive buffer
1	TXBUFF		transmit buffer
		0	No need to clear the transmit buffer
		1	clear the transmit buffer

7-2	-	-	reserved
-----	---	---	----------

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.19. CANFD_GetFIFOStatus

Description

This function is used to get the receive/transmit data buffer overflow state of the CAN port on a specified PISO-CANFD series board.

Syntax

```
C#
.....
Int16 CANFD_GetFIFOStatus (
    Byte BoardNo,
    Byte Port,
    out Byte FIFOState
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3
- 4: CAN port 4.

FIFOState

[out] The receive/ transmit data buffer overflow state of the CAN port.

Bit	Symbol	Value	Description
0	RXO		receive buffer overflow flag
		0	receive buffer not overflow
		1	receive buffer overflow
1	TXO		transmit buffer overflow flag
		0	transmit buffer not overflow

		1	transmit buffer overflow
7-2	-	-	reserved

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.20. CANFD_GetCANStatus

Description

This function is used to retrieve the CAN bus error status and transmit/receive error counter of the CAN port on a specified PISO-CANFD series board.

Syntax

```
C#
.....
Int16 CANFD_CANGetStatus (
    Byte BoardNo,
    Byte Port,
    out UInt16 CANStatus,
    out Byte TxErrorCount
    out Byte RxErrorCount
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3
- 4: CAN port 4.

CANStatus

[out] Indicates the address of a variable used to retrieve the status register value for the CAN controller.

Bit	Symbol	Value	Description
0	EWARN		Transmitter or Receiver is in Error Warning State bit
		0	Transmitter or Receiver is not in Error Warning State

		1	Transmitter or Receiver is in Error Warning State
1	RXWARN		Receiver in Error Warning State bit
		0	Receiver not in Error Warning State (REC ≤ 95)
		1	Receiver in Error Warning State (128 > REC > 95)
2	TXWARN		Transmitter in Error Warning State bit
		0	Transmitter not in Error Warning State (TEC ≤ 95)
		1	Transmitter in Error Warning State (128 > TEC > 95)
3	RXBP		Receiver in Error Passive State bit
		0	Receiver not in Error Passive State (REC ≤ 127)
		1	Receiver in Error Passive State (REC > 127)
4	TXBP		Transmitter in Error Passive State bit
		0	Transmitter in Error Passive State (TEC ≤ 127)
		1	Transmitter in Error Passive State (TEC > 127)
5	TXBO		Transmitter in Bus Off State bit
		0	Transmitter not in Bus Off State
		1	Transmitter in Bus Off State (TEC > 255). In Configuration mode, TXBO is set, since the CAN module is not on the bus
15-6	-	-	reserved

TxErrorCount

[out] Indicates the address of a variable used to retrieve the transmit error counter value of the CAN controller.

RxErrorCount

[out] Indicates the address of a variable used to retrieve the receiver error counter value of the CAN controller.

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.2.21. CANFD_GetBUSDiagnostic

Description

This function is used to retrieve the CAN bus error status (including separate error counters for receive/transmit and for nominal/data bit rates on BUSDiag0 and track of the kind of error that occurred since the last reading BUSDiag1) of the CAN port on a specified PISO-CANFD series board.

Syntax

```
C#
.....
Int16 CANFD_GetBUSDiagnostic (
    Byte BoardNo,
    Byte Port,
    out UInt32 BUSDiag0,
    out UInt32 BUSDiag1
);
```

Parameters

BoardNo

[in] Specifies the number of the PISO-CANFD series board to be read. The valid range is 0 to 15.

Port

[in] Specifies the number of the CAN port, where:

- 1: CAN port 1
- 2: CAN port 2.
- 3: CAN port 3
- 4: CAN port 4.

BUSDiag0

[out] BUSDIAG0 contains separate error counters for receive/transmit and for nominal/data bit rates. They are simply incremented by one on every error.

Bit	Symbol	Description
7-0	NRERRCNT	Nominal Bit Rate Receive Error Counter bits
15-8	NTERRCNT	Nominal Bit Rate Transmit Error Counter bits

23-16	DRERRCNT	Data Bit Rate Receive Error Counter bits
31-24	DTERRCNT	Data Bit Rate Transmit Error Counter bits

BUSDiag1

[out] BUSDIAG1 keeps track of the kind of error that occurred since the last clearing of the register. The register also contains the error-free message counter. The flags and the counter are cleared after reading the register

Bit	Symbol	Description
15-0	EFMSGCNT	Error Free Message Counter bits
16	NBIT0ERR	During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a dominant level (data or identifier bit logical value '0'), but the monitored bus value was recessive.
17	NBIT1ERR	During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value '1'), but the monitored bus value was dominant.
18	NACKERR	Transmitted message was not acknowledged
19	NFORMERR	A fixed format part of a received frame has the wrong format
20	NSTUFERR	More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed
21	NCRCERR	The CRC check sum of a received message was incorrect. The CRC of an incoming message does not match with the CRC calculated from the received data.
22	-	Reserved, read as '0'
23	TXBOERR	Device went to bus-off (and auto-recovered).
24	DBIT0ERR	Same as for nominal bit rate (see above).
25	DBIT1ERR	Same as for nominal bit rate (see above).
26	-	Reserved, read as '0'
27	DFORMERR	Same as for nominal bit rate (see above).
28	DSTUFERR	Same as for nominal bit rate (see above).
29	DCRCERR	Same as for nominal bit rate (see above).
30	ESI	ESI flag of a received CAN FD message was set.
31	DLCMM	transmission or reception DLC mismatch bit

Return Value

If the function succeeds, the return value will be 0.

If the function fails, refer to Section 3.6 Error Code Definitions.

3.6. Error Code Definitions

The following are the definitions for the error codes that may be encountered while operating the PISO-CANFD.

Error Code (hexadecimal)	Error ID	Error Description
0x000	ERR_NO_ERR	No error
0x001	ERR_INIT_DRIVER_ERROR	An error occurred while initializing the driver
0x002	ERR_COMM_DRIVER_ERROR	An error occurred while communicating with the driver
0x003	ERR_DRIVER_UNINIT_ERROR	The driver is not initialized
0x004	ERR_ACTIVE_BOARD_ERROR	An error occurred while activating the board
0x005	ERR_BOARD_ALREADY_ACTIVE_ERROR	The board has already been activated
0x006	ERR_BOARD_UNACTIVE_ERROR	The board is not activated
0x007	ERR_BOARD_NUMBER_ERROR	The board value exceeds the valid range
0x008	ERR_PORT_NUMBER_ERROR	The CAN port value exceeds the valid range
0x009	ERR_PORT_UNINIT_ERROR	The CAN port is not initialized.
0x020	ERR_FLT_FRAME_FORMAT_ERROR	CAN filter format setting must be standard identifier(0) or extended identifier(1)
0x021	ERR_FLT_ACC_ACM_INCONSISTENCY_ERROR	CAN filter setting of inMask and inAcceptance arguments should be "inAcceptance & inMask == inAcceptance"
0x022	ERR_FLT_STD_ACC_TOOLARGE_ERROR	The standard identifier of CAN acceptance filter setting large than 0x7FF
0x023	ERR_FLT_EXT_ACC_TOOLARGE_ERROR	The extended identifier of CAN acceptance filter setting large than 0x1FFFFFFF.
0x024	ERR_FLT_STD_ACM_TOOLARGE_ERROR	The standard identifier of

		CAN mask filter setting large than 0x7FF
0x025	ERR_FLT_EXT_ACM_TOOLARGE_ERROR	The extended identifier of CAN mask filter setting large than 0x1FFFFFFF.
0x030	ERR_INVALID_OPMODE_ERROR	CAN operation mode setting is not support
0x031	ERR_SOFTBUFF_IS_EMPTY	CAN receive buffer is empty
0x032	ERR_NORM_BITRATE_NOT_SUPP_ERROR	CAN normal bit rate setting is not support
0x033	ERR_DATA_BITRATE_NOT_SUPP_ERROR	CAN data bit rate setting is not support
0x034	ERR_DATABR_LESS_THAN_NORMBR_ERROR	CAN data bit rate setting is less than normal bit rate setting
0x035	ERR_INVALID_NORMAL_SAMPLEPOINT	CAN normal baudrate sample point setting is not support
0x036	ERR_INVALID_DATA_SAMPLEPOINT	CAN data baudrate sample point setting is not support
0x101	ERR_SYS_BOARD_ALREADY_ACTIVE	The board has already been activated.
0x102	ERR_SYS_PORT_NUMBER_ERROR	The Port value is invalid..
0x103	ERR_SYS_INSTALL_IRQ_ERROR	An error occurred while installing the system interrupt
0x110	ERR_SYS_CAN_RESET_ERROR	Reset CAN chip error
0x111	ERR_SYS_CAN_OPMODE_ERROR	CAN chip operation mode error
0x112	ERR_SYS_REQ_CONFIG_MODE_TIMEOUT	Change CAN chip to configuration mode timeout
0x113	ERR_SYS_CHANGE_MODE_TIMEOUT	Change CAN chip operation mode timeout
0x114	ERR_SYS_SPI_FULLSPEED_RAMTEST_ERROR	Test CAN chip RAM data error
0x115	ERR_SYS_CAN_FilterIdx_OutOfRange	CAN filter index setting is out of range.
0x120	ERR_SYS_CAN_TRANSMIT_ERR_DETECTED	Detect an error when

		transmitting a CAN message
0x121	ERR_SYS_SOFTBUFF_IS_EMPTY	CAN receive buffer of driver is empty
0x122	ERR_SYS_SOFTBUFF_IS_FULL	CAN receive buffer of driver is full
0x123	ERR_SYS_HARDWAREBUFF_IS_EMPTY	CAN receive buffer of CAN chip is empty
0x124	ERR_SYS_HARDWAREBUFF_IS_FULL	CAN transmit buffer of CAN chip is full

4. Appendix

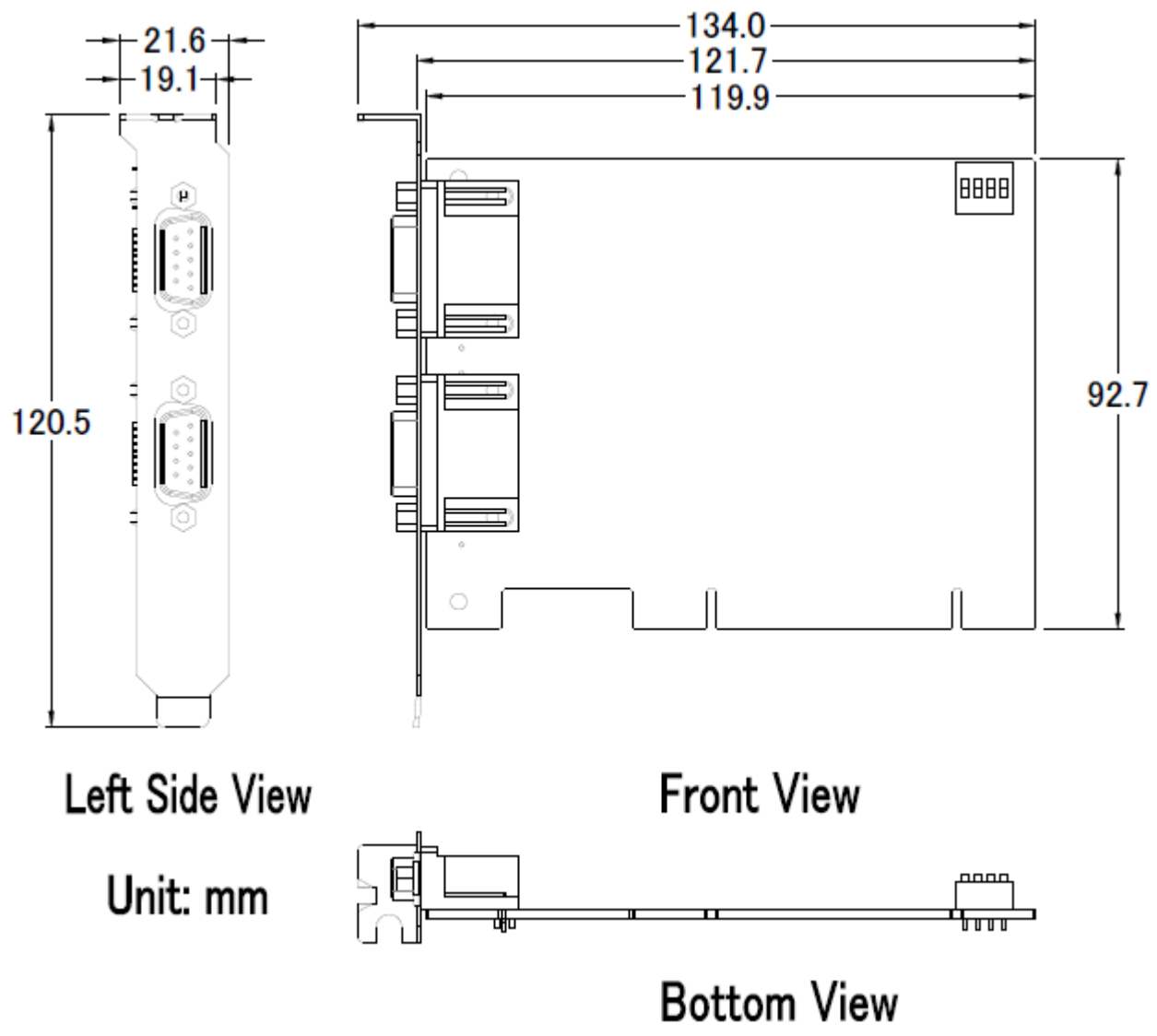
4.1. Revision History

This chapter provides revision history information to this document.
The table below shows the revision history.

Revision	Date	Description
1.0.0	2021/01/31	Initial issue

4.2. Dimensions

4.2.1. PISO-CAN200U-FD-D / PISO-CAN400U-FD-D



4.2.2. PISO-CAN200U-FD-T / PISO-CAN400U-FD-T

