

PCI-58XX

Isolated PCI CAN Interface Card Datasheet

Contents

Contents.....	1
Chapter 1 Product Overview	2
1.1 Product Overview.....	2
1.2 Product Appearance	2
1.3 Parameters	3
1.4 Typical applications	3
1.5 Ordering Information	4
1.6 Product Sales list	4
1.7 Support Information	4
1.8 Shipping	4
Chapter 2 Hardware Description	5
2.1 Mechanical Size	5
2.2 Interface Description.....	6
2.2.1 CAN Interface.....	6
Chapter 3 Hardware Installation.....	8
3.1 Installation Procedure.....	8
Chapter 4 Software development.....	9
4.1 Data Structure of Library	9
4.1.1 PCI58XX_INIT_CONFIG.....	9
4.1.2 PCI58XX_CAN_OBJ	10
4.2 Function description	11
4.3 Interface library function using flow.....	13
Appendix A: SJA1000 standard Baud rate.....	14
(Oscillator Frequency=16MHz)	14
Appendix B: Product Catalogue	错误!未定义书签。

Chapter 1 Product Overview

1.1 Product Overview

The PCI-58XX is a Controller Area Network (CAN) interface card. It supports one or two CAN interface that can run independently at the same time. The built-in CAN controller of this card is Philips SJA1000T, which provides bus arbitration and error detection with auto correction and re-transmission function.

1.2 Product Appearance

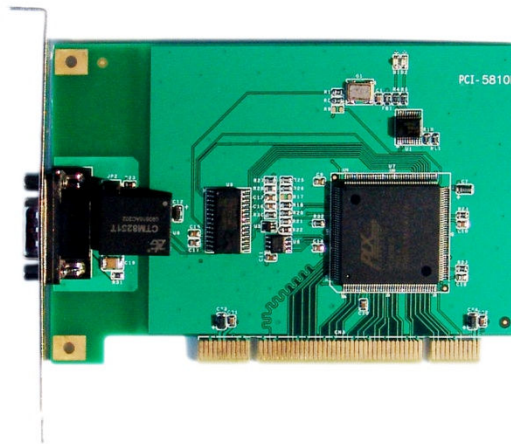


Figure 1-2-1 Appearance of the Single-port Isolated PCI CAN Interface Card (PCI-5810I)

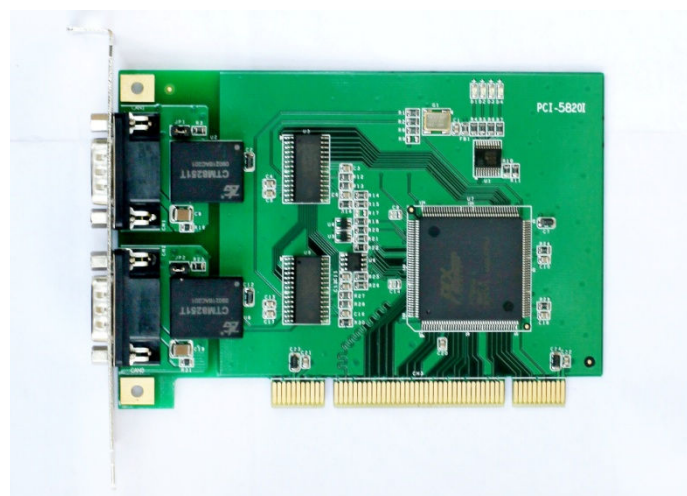


Figure 1-2-2 Appearance of the Dual-port Isolated PCI CAN Interface Card (PCI-5820I)

PCI-58XX

Isolated PCI CAN Interface Card

Product Datasheet

1.3 Parameters

- **Passive CAN Interface for PCI slots**
- **One, two CAN channels with Controller NXP SJA1000**
- **Baud rates up to 1MBaud, 82C251 Transceiver**
- **CAN is connected via a 9-pin SUB-D as defined by the CiA DS102-1 standard**
- **CAN 2.0A (11-bit ID standard frames) and 2.0B (29-bit ID extended frames)**
- **Development kits for Windows 2000/XP/Vista**
- **Examples of Visual Basic 6.0, Visual C++ 6.0, C++builder 6.0, Delphi7.0, Labview are available**
- **Max data flow 4000 fps (extend frame)**
- **Adopts electrical isolation, the isolation voltage is : 2500Vrms;**
- **Operating temperature: -20 to +70**

1.4 Typical applications

- **CAN-bus network diagnosis and test**
- **Auto electronic applications**
- **Electric power communication network**
- **Industrial control devices**
- **High-speed and large data communications**

PCI-58XX

Isolated PCI CAN Interface Card

Product Datasheet

1.5 Ordering Information

Part Number	Operating temperature	CAN Channel	Interface
PCI-5810I	-20°C ~+70°C	1	DB9
PCI-5820I	-20°C ~+70°C	2	DB9

1.6 Product Sales list

- **PCI-58XX PCI CAN Card ;**
- **CD-ROM.(Datasheet, Drivers, DLL, PCI-58XX Tester software, VB, VB2003, VC, C++Builder, Delphi, Labview examples)**

1.7 Support Information

Technical Support Mail: sales@buenoptic.net

Web site: <http://www.buenoptic.net>

1.8 Shipping

We will choose EMS, UPS, FEDEX, DHL or TNT Express to deliver your order. It takes 3-9 business days to most places. Usually we ship your item out in 48 hours next to payment cleared, and then email you the tracking number.

Chapter 2 Hardware Description

2.1 Mechanical Size

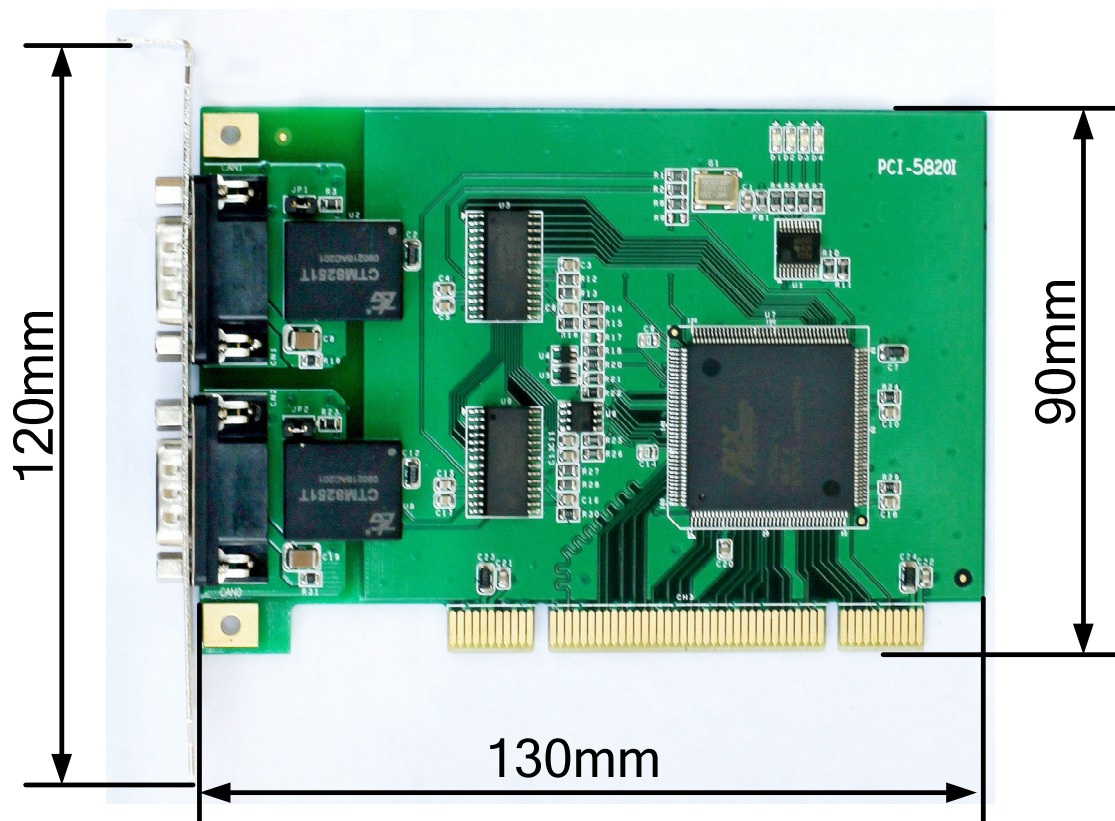


Figure 2-1-1 Mechanical size of the PCI58XX PCI CAN Card

2.2 Interface Description

2.2.1 CAN Interface

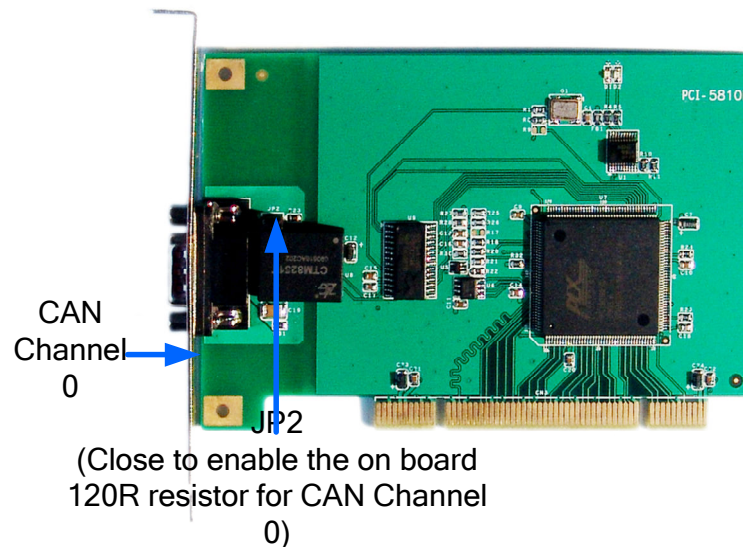


Figure 2-2-1-1 CAN Interface (PCI-5810I)

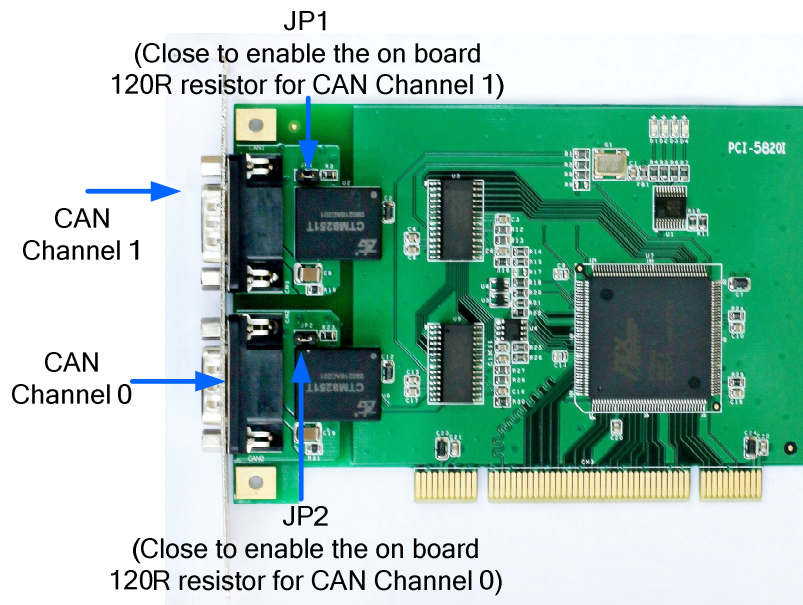


Figure 2-2-1-2 CAN Interface (PCI-5820I)

PCI-5820I integrates two CAN-bus channels, while PCI-5810I integrates one. The pin signal definitions Table 2-2-1-1.

PCI-58XX

Table 2-2-1-1 Description of the DB9 hardware interface for CAN BUS

CAN channel 0 and channel 1 are the same

Pin	Signal	Description
1	N.C.	
2	CAN_L	CAN_L of the CAN BUS
3	CAN_GND	Reference GND
4	N.C.	
5	CAN_SHIELD	Shield Cable
6	CAN_GND	Reference GND
7	CAN_H	CAN_H of the CAN BUS
8	N.C.	
9	N.C.	

Chapter 3 Hardware Installation

3.1 Installation Procedure

Do the following to install the PCI-58XX PCI CAN card in the computer:

Attention! Electrostatic discharge (ESD) can damage or destroy components on the PCI CAN card. Take precautions to avoid ESD when handling the card.

1. Shut down the computer.
2. Disconnect the computer from the power supply.
3. Open the computer's casing.
4. Insert the PCI-58XX PCI CAN card into an empty PCI slot. For details please refer to the documentation of the computer.
5. Close the computer's casing.
6. Reconnect the power supply of the computer.

Do the following to complete the initialization:

1. Turn on the computer and start Windows. Make sure that you are logged in as user with administrator privileges. Windows notifies that new hardware has been detected.
2. A Wizard dialog box appears. Follow its instructions. Deny the search for driver software at Windows update and select the driver's directory during the procedure. Next, the drivers are found and installed by Windows.
3. After the initialization process is finished successfully you can find the entry "PCI-58XX PCICAN Communication Card" in the branch "PCI CAN" of the Windows Device Manager.

Chapter 4 Software development

If users intend to make a program for their own application, they need to read following descriptions very carefully, and refer the demo source code.

Develop files include PCI58XX.h, PCI58XX.lib (For VC) , PCI58XXbc.lib (For BC), PCI58XX.dll.

We provides examples for **VB**, **VB2003**, **VC**, **C++Builder**, **Delphi**, **Labview** which make it convenient for user to develop programs.

4.1 Data Structure of Library

4.1.1 PCI58XX_INIT_CONFIG

```
typedef struct _ PCI58XX_INIT_CONFIG{  
    DWORD    AccCode;  
    DWORD    AccMask;  
    DWORD    Reserved;  
    UCHAR    Filter;  
    UCHAR    Timing0;  
    UCHAR    Timing1;  
    UCHAR    Mode;  
} PCI58XX_INIT_CONFIG,*P_PCI58XX_INIT_CONFIG;
```

AccCode acceptance code for filter

AccMask mask code for filter

Reserved not used

Filter filter mode , single or double

Timing0 timer0 (BTR0)

Timing1 timer1 (BTR1)

Mode work mode 0: normal work, 1: listen only

Timing0 and remark Timing1 is used for setting CAN baud rate. The following table is about setting of 15 kinds of common baud rates.

Table 4-1-1-1 Standard Time0 and Timer1 value for SJA100T

CAN Baud rate	Timer0	Timer1
5Kbps	0xBF	0xFF
10Kbps	0x31	0x1C
20Kbps	0x18	0x1C
40Kbps	0x87	0xFF
50Kbps	0x09	0x1C
80Kbps	0x83	0Xff
100Kbps	0x04	0x1C
125Kbps	0x03	0x1C
200Kbps	0x81	0xFA
250Kbps	0x01	0x1C
400Kbps	0x80	0xFA
500Kbps	0x00	0x1C
666Kbps	0x80	0xB6
800Kbps	0x00	0x16
1000Kbps	0x00	0x14

4.1.2 PCI58XX_CAN_OBJ

```
typedef struct _PCI58XX_CAN_OBJ{
    DWORD    ID;
    UCHAR    SendType;
    UCHAR    ExternFlag;
    UCHAR    RemoteFlag;
    UCHAR    DataLen;
    UCHAR    Data[8];
} PCI58XX_CAN_OBJ,*P_PCI58XX_CAN_OBJ;
```

ID packet ID, 4 bytes

SendType 0: Normal Send,1 :self reception

RemoteFlag remote frame or not

ExternFlag extended frame or not

DataLen data length(<=8), it is the length of data

Data data of packet

4.2 Function description

[1] [Open the device.](#)

BOOL __stdcall PCI58XX_OpenDevice(DWORD DevIndex);
DevIndex Device index.0: The first device.1: The second device.
Return value 1: Success, 0: Fail

[2] [Close the device](#)

BOOL __stdcall PCI58XX_CloseDevice(DWORD DevIndex);
DevIndex Device index.0: The first device.1: The second device.
Return value 1: Success, 0: Fail

[3] [Initialize can.](#)

BOOL __stdcall PCI58XX_InitCan(DWORD DevIndex,DWORD CANIndex,
P_PCI58XX_INIT_CONFIG InitConfig);
DevIndex Device index.0: The first device.1: The second device.
CANIndex Can channel.0: The first channel.1: The second channel
InitConfig Init parameters structure.

InitConfig->AccCode	AccCode corresponds to four registers in SJA1000. ACR0=((InitConfig->AccCode)>>24)&0xFF
InitConfig->AccMask	AMR0=((InitConfig->AccMask)>>24)&0xFF
InitConfig->Reserved	reserved
InitConfig->Filter	Filter mode, 0- single filter, 1-dual filter
InitConfig-> timer0	Baud rate timer 0
InitConfig-> timer1	Baud rate timer 1
InitConfig->Mode	0:Normal mode.1:Listen only

Return value 1: Success, 0: Fail

[4] [Reset can.](#)

BOOL __stdcall PCI58XX_ResetCan(DWORD DevIndex , DWORD CANIndex);
DevIndex Device index.0: The first device.1: The second device.
CANIndex Can channel.0: The first channel.1: The second channel
Return value 1: Success, 0: Fail

[5] [Send can packet.](#)

BOOL __stdcall PCI58XX_Transmit(DWORD DevIndex , DWORD CANIndex,
P_PCI58XX_CAN_OBJ *pSend);

DevIndex Device index.0: The first device.1: The second device.

CANIndex Can channel.0: The first channel.1: The second channel

pSend Packet to Send.

Return value 1: Success, 0: Fail

[6] [Receive can packet.](#)

DWORD __stdcall PCI58XX_Receive(DWORD DevIndex , DWORD CANIndex,
PPCI58XX_CAN_OBJ pReceive , DWORD Len , DWORD WaitTime);

DevIndex Device index.0: The first device.1: The second device.

CANIndex Can channel.0: The first channel.1: The second channel

pReceive Receive packet.

Len Number of packets you need to read.

WaitTime Time out.

Return value Actually packets number returned.

[7] [Get the number packet in the internal buffer](#)

DWORD __stdcall PCI58XX_GetReceiveNum(DWORD DevIndex,DWORD
CANIndex);

DevIndex Device index.0: The first device.1: The second device.

CANIndex Can channel.0: The first channel.1: The second channel

Return value number of the packets in the internal buffer

[8] [Clear the internal buffer.](#)

BOOL __stdcall PCI58XX_ClearBuffer(DWORD DevIndex,DWORD CANIndex);

DevIndex Device index.0: The first device.1: The second device.

CANIndex Can channel.0: The first channel.1: The second channel

Return value 1: Success, 0: Fail

4.3 Interface library function using flow

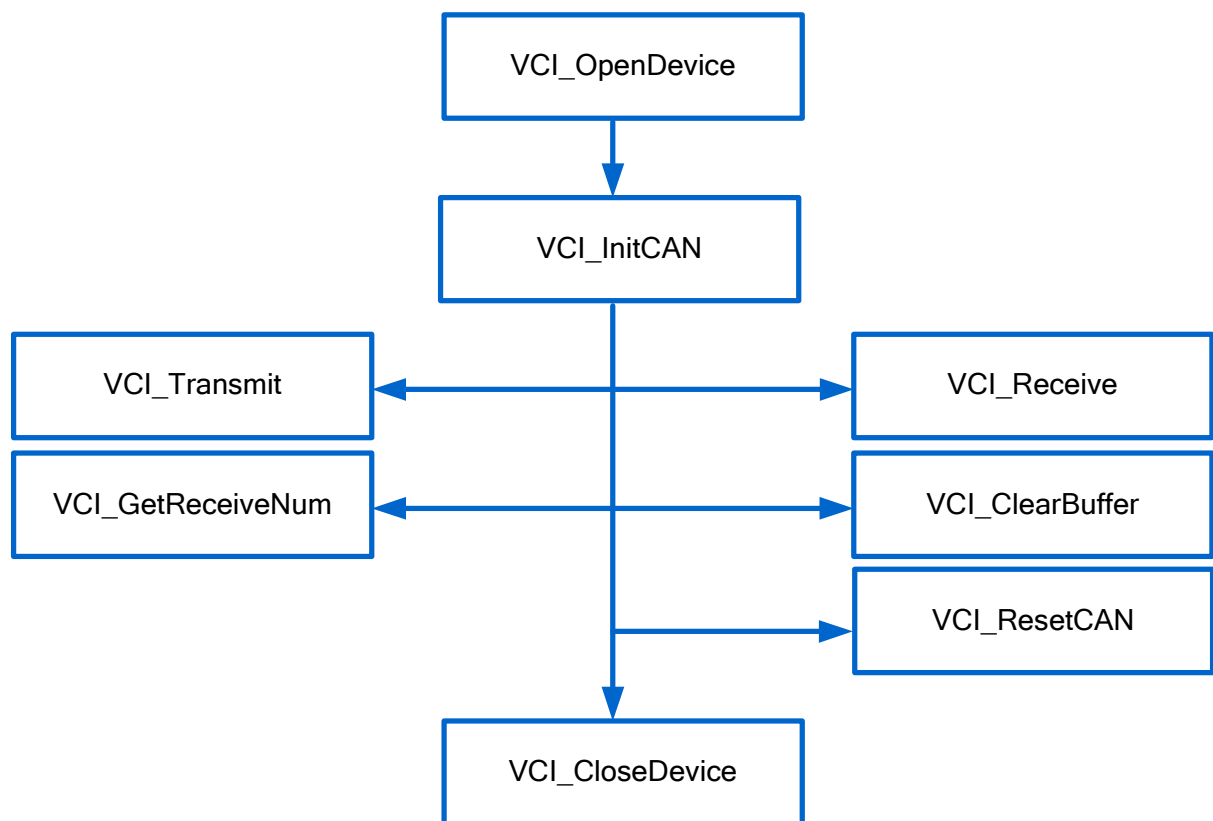


Figure 4-3-1 Calling flow of the interface function

Appendix A: SJA1000 standard Baud rate

(Oscillator Frequency=16MHz)

CAN Baud rate	BTR0 or Timer0(Hex)	BTR1 or Timer1(Hex)
5Kbps	0xBF	0xFF
10Kbps	0x31	0x1C
20Kbps	0x18	0x1C
40Kbps	0x87	0xFF
50Kbps	0x09	0x1C
80Kbps	0x83	0Xff
100Kbps	0x04	0x1C
125Kbps	0x03	0x1C
200Kbps	0x81	0xFA
250Kbps	0x01	0x1C
400Kbps	0x80	0xFA
500Kbps	0x00	0x1C
666Kbps	0x80	0xB6
800Kbps	0x00	0x16
1000Kbps	0x00	0x14