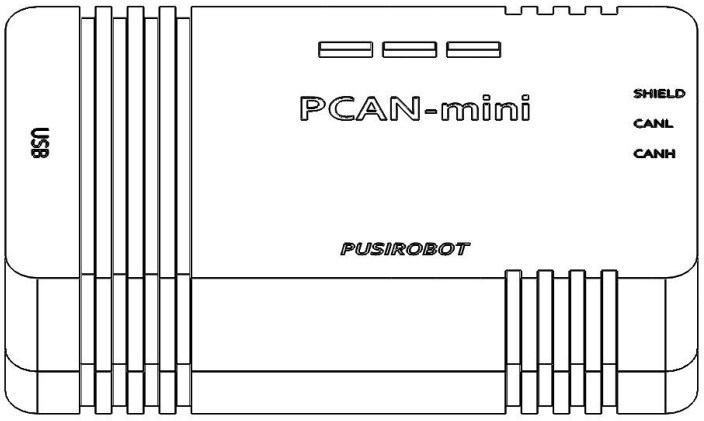
PUSIROBOT

CQPUSI ROBOT CONTROL SYSTEM

User Manual

PCAN_MINI Series

CAN Adapter



1. Version Control

1） Update Records

Date	Author	Version	Remark
2023-11-30	Sen	V0.1.0	Initial

Catalog

1 Introduction

1.1 Statement of intellectual property right

PCAN_MINI series CAN analyzers have applied for the following national patents:

- The CAN analyzer scheme and method have applied for invention patent protection.
- The CAN analyzer circuit has applied for utility model patent protection.
- The appearance of the CAN analyzer has applied for appearance patent protection.

The PCAN_MINI series of CAN analyzers have firmware code embedded in them, and any attempt to breach the protection of the firmware code can be considered a violation of the intellectual property protection laws and regulations. CQPUSI shall have the right to sue under the Act to stop such conduct if such conduct results in the acquisition of software or other intellectual property protected work without authorization from CQPUSI.

1.2 Disclaimer

The using method of the device and other content in the description of this manual is only used to provide convenience for you, and may be update in future version. To ensure the application conforms to the technical specifications is the responsibility of your own. CQPUSI does not make any form of statement or guarantee to the information, which include but not limited to usage, quality, performance, merchantability or applicability of specific purpose. CQPUSI is not responsible for these information and the consequences result caused by such information. If the CQPUSI device is used for life support and/or life safety applications, all risks are borne by the buyer. The buyer agrees to protect the CQPUSI from legal liability and compensation for any injury, claim, lawsuit or loss caused by the application.

2 Overview

2.1 General Description

PCAN_MINI is a miniature USB-CAN adapter in a small size which is easy to carry, suitable for mobile applications, takes less space on the device. Different operating systems (Windows, Linux, Raspberry Pi, etc.) are supported, and you can connect to the CAN bus via the PCAN serial tool or PUSICAN, the companion software for our stepper motor controllers.

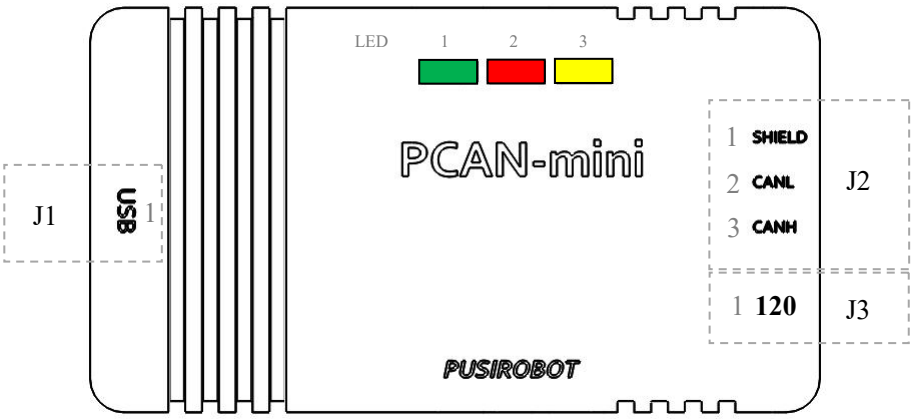
The adapter uses a highly integrated fully isolated chip with 2500V rms signal and 1500VDC power isolation, and has multiple hardware protection circuitry to protect your PC from electrical interference and static electricity in the operating environment, making it suitable for industrial debugging or engineering applications.

2.2 Features

- ✓ USB interface for power supply and communication
- ✓ Indicator for power and communication
- ✓ Optional DIP switch of terminal resistance
- ✓ CAN galvanic isolation voltage 1500VDC
- ✓ USB connection (Full-Speed mode, compatible with USB 1.1, USB 2.0, and USB 3.0)
- ✓ Compliant with CAN specifications 2.0A (11-bit ID) and 2.0B (29-bit ID) maximum baud rate is 1Mbps

3 Connector Description

3.1 Terminal port location



3.2 Signal connection J1

Pin	1
Designator	USB

Description:

USB: Type-B interface

3.3 Signal connection J2

Pin	1	2	3
Designator	SHIELD	CANL	CANH

Description:

SHIELD: Signal isolation of CAN bus;

CANL: Low-level signal cable of CAN bus;

CANH: High-level signal cable of CAN bus;

3.4 DIP Switch J3

Pin	1
Designator	120

Description:

120: Terminal resistance of CAN bus;

Note: The high-speed CAN bus (ISO 11898-2) must be equipped with 120 ohm terminal resistance at both ends of the bus to prevent the reflection of interfering signals and ensure that the transceiver connected to the CAN node (CAN interface, control device) is working properly.

3.5 LED Light

Indicator 1 (Green) : Power supply voltage exists, solid when powered on

Indicator 2 (Red) : TX signal indicator, flashing when signal is sending

Indicator 3 (Yellow) : RX signal indicator, flashing when signal is receiving

4 Driver installation

4.1 Installation steps

1. Unzip the driver installation package obtained from the vendor

2. Configure the installation options according to the default settings or according to your needs, then install it

4.2 Connect adapter

1. Connect the adapter to the USB port of the computer, the PC detects the new hardware and completes the driver installation.

5 Secondary development

Provide secondary development data packages

This API provides the ability to link your own programs to CAN and CAN FD interfaces via PCAN, supporting operating systems Windows 10, 8.1, 7 (32/64-bit), Windows CE 6, Linux (32/64-bit), Windows 10, 8.1, 7 (32/64-bit).

5.1 SocketCAN Secondary development （By API system （C programming language））

The application first sets up access to the CAN interface by initializing a socket (much like the situation in TCP/IP communication) and then binding that socket to one interface (or all interfaces, if required by the application). Once bound, the socket can be used for read, write, and more.

Before the secondary development, you need to install the can_dev module and configure the CAN bus baud rate, and then enable it. For example:

```
$ modprobe can_dev
$ modprobe can
$ modprobe can_raw
$ sudo ip link set can0 type can bitrate 500000
$ sudo ip link set up can0
```

The following code snippet is a example of the SocketCAN API, which uses the original interface to send packets:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <net/if.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <linux/can.h>
#include <linux/can/raw.h>
int main(void)
{
int s;
int nbytes;
struct sockaddr_can addr;
struct can_frame frame;
struct ifreq ifr;
```

```
const char *ifname = "vcan0";
if((s = socket(PF_CAN, SOCK_RAW, CAN_RAW)) < 0) {
perror("Error while opening socket");
return -1;
}
strcpy(ifr.ifr_name, ifname);
ioctl(s, SIOCGIFINDEX, &ifr);
addr.can_family = AF_CAN;
addr.can_ifindex = ifr.ifr_ifindex;
printf("%s at index %d\n", ifname, ifr.ifr_ifindex);
if(bind(s, (struct sockaddr *)&addr, sizeof(addr)) < 0) {
perror("Error in socket bind");
return -2;
}
frame.can_id = 0x123;
frame.can_dlc = 2;
frame.data[0] = 0x11;
frame.data[1] = 0x22;
nbytes = write(s, &frame, sizeof(struct can_frame));
printf("Wrote %d bytes\n", nbytes);
return 0;
}
```

5.2 Using Secondary development（Windows)

The simplest communication of CAN using the API, the following example demonstrates the basic steps of initializing the API, sending CAN messages, receiving CAN messages, and shutting down the API (the actual application requires modifying the PCAN channel, baud rate, and sent/received CAN messages).

```
#include <iostream>
#include <windows.h>
#include "PCANBasic.h"

int main()
{
    TPCANStatus status;
    TPCANHandle hnd;
    TPCANMsg canMsg;

    // 初始化 API
    status = CAN_Initialize(PCAN_USBBUS1, PCAN_BAUD_500K, 0, 0, 0);
    if (status != PCAN_ERROR_OK)
    {
```

```
        std::cout << "CAN_InitializeError：" << status << std::endl;
        return 1;
    }

    // 准备 CAN 消息
    canMsg.ID = 0x123;
    canMsg.LEN = 8;
    canMsg.MSGTYPE = PCAN_MESSAGE_STANDARD;
    for (int i = 0; i < 8; i++)
        canMsg.DATA[i] = i;

    // 发送 CAN 消息
    status = CAN_Write(PCAN_USBBUS1, &canMsg);
    if (status != PCAN_ERROR_OK)
    {
        std::cout << "CAN_WriteError：" << status << std::endl;
        return 1;
    }

    // 接收 CAN 消息
    status = CAN_Read(PCAN_USBBUS1, &canMsg, nullptr);
    if (status == PCAN_ERROR_OK)
    {
        std::cout << "CAN_ReadID: " << std::hex << canMsg.ID << " Read_data: ";
        for (int i = 0; i < canMsg.LEN; i++)

            std::cout << std::hex << static_cast<int>(canMsg.DATA[i]) << " ";
        std::cout << std::dec << std::endl;
    }
    else
    {
        std::cout << "ReadError：" << status << std::endl;
        return 1;
    }

    // 关闭  API
    status = CAN_Uninitialize(PCAN_USBBUS1);
    if (status != PCAN_ERROR_OK)
    {
        std::cout << "CAN_UninitializeError：" << status << std::endl;
        return 1;
    }

    return 0;
```

}

5.3 Using Python

Python 3.3 and later versions have updated support for SocketCAN. The open-source library python-can provides SocketCAN support for python 2 and python 3. For more information, please refer to the python-can official documentation.

6 Dimensions（mm）