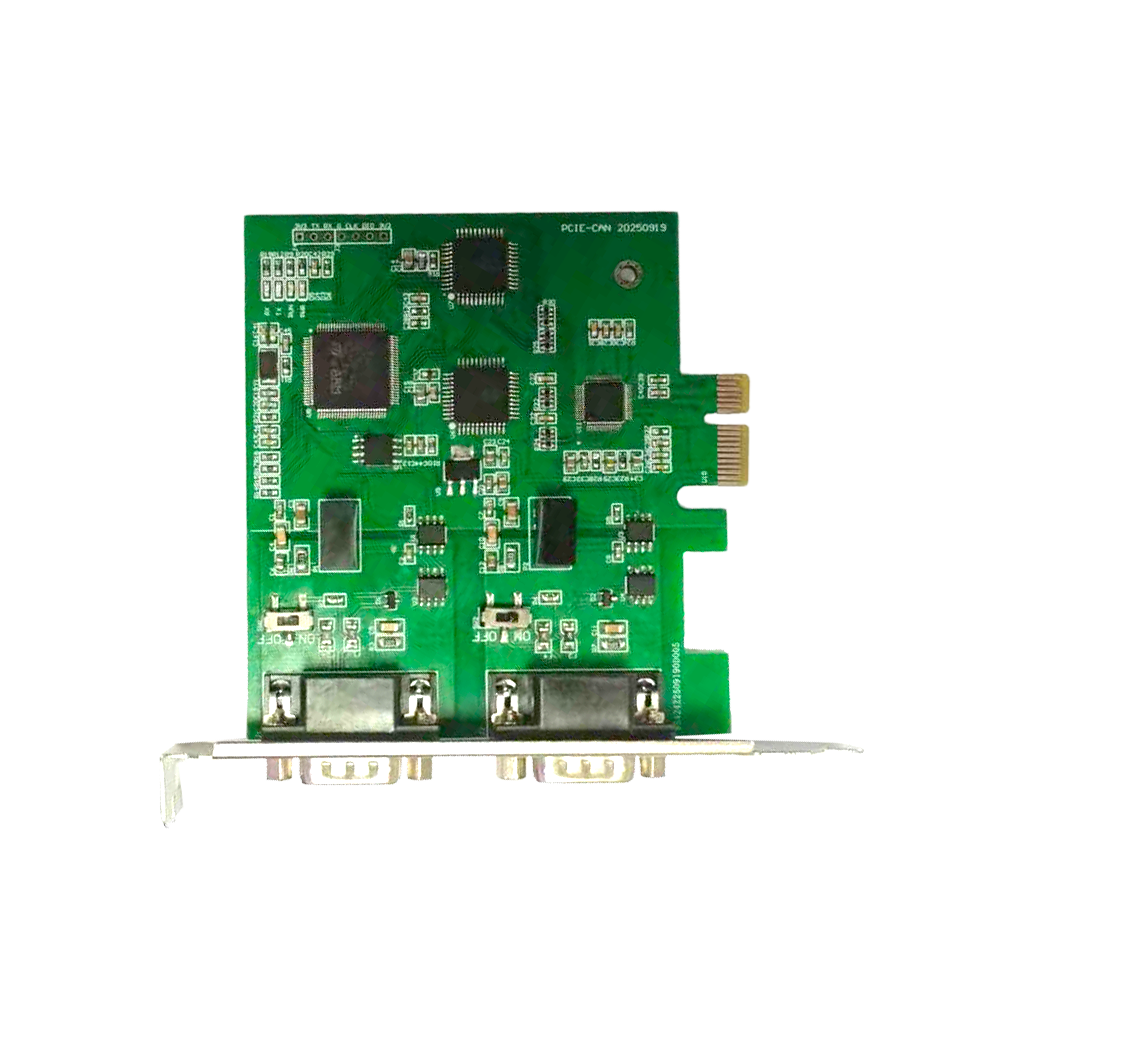
**CAN（FD）接口卡库函数说明**

**型号：SG-PCIe-CAN(FD)-200T**

****

天津滨海新区三格电子科技有限公司

[www.tj-sange.com](http://www.tj-sange.com/)

# 版本信息

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 版本号 | 修改内容 | 备注 |
| 2025/09/16 | v1.0.0 | 建立 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 目录

[版本信息 2](#_Toc23711)

[目录 3](#_Toc21034)

[一、测试软件使用说明 4](#_Toc26191)

[1.1 设备操作 4](#_Toc24130)

[1.2 快捷工具栏说明 5](#_Toc24886)

[1.3 报文显示窗口 5](#_Toc16421)

[1.4 报文发送窗口 5](#_Toc13157)

[二、库函数简介 5](#_Toc4905)

[2.1 库函数使用简介 5](#_Toc19011)

[2.2 库函数调用流程 6](#_Toc22776)

[三、接口卡参数 6](#_Toc11068)

[3.1 型号参数 6](#_Toc10740)

[3.2 库函数数据结构说明 6](#_Toc15156)

[3.2.1 CAN初始化数据类型 6](#_Toc21867)

[3.2.2 CAN信息帧数据类型 7](#_Toc31309)

[3.2.3 CAN 滤波器数据类型 8](#_Toc22726)

[3.3 接口函数说明 8](#_Toc7921)

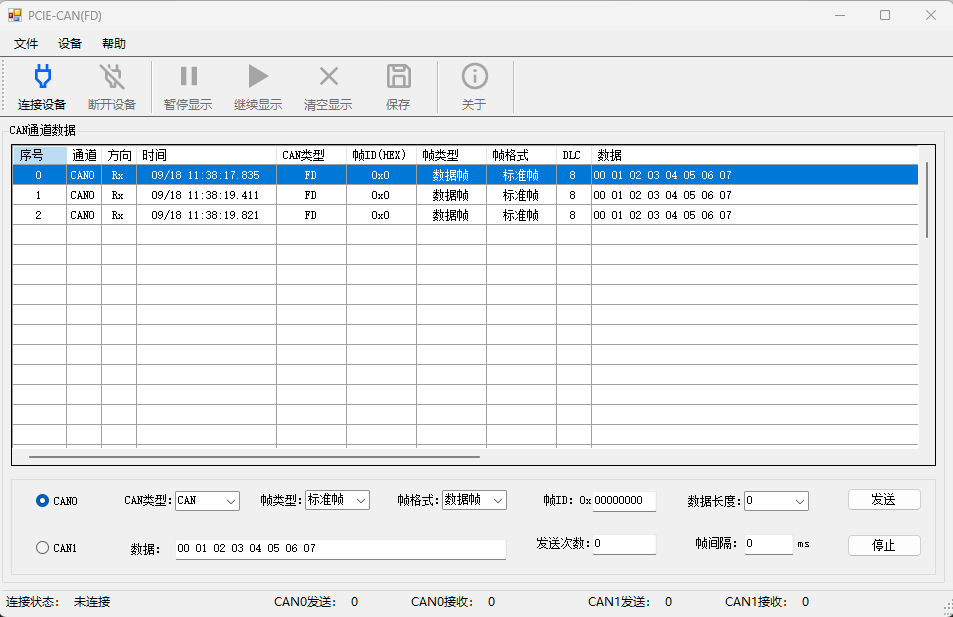
[四、接口库使用方法 11](#_Toc3290)

[4.1 VC调用动态库方法 11](#_Toc4692)

[五、售后及联系方式 12](#_Toc8953)

# 一、测试软件使用说明

PCIe-CAN(FD)测试软件，是针对我司PCIe、MINI PCIe接口的CAN(FD)通讯卡进行测试，从而熟悉板卡性能，主界面如下：



## 设备操作

点击连接设备，在弹出窗口中设置can通讯CAN通讯参数，完成后点击“确认”，即可打开设备，如下：



## 快捷工具栏说明



连接设备：用于连接待测设备。

断开设备：用于断开待测设备。

暂停显示：用于暂停报文显示窗口接收CAN报文。

继续显示：恢复显示状态。

清空显示：清空报文显示窗口中的所有报文。

保存：保存报文显示窗口的所有报文。

关于：公司及售后详情。

## 报文显示窗口



## 报文发送窗口



# 二、库函数简介

## 2.1 库函数使用简介

我司推出PCIe、MINI PCIe接口的CAN（FD）接口卡，以满足CAN（FD）市场发展的需求。

CAN（FD）接口卡支持Windows、Linux下进行二次开发，以动态链接库的方式驱动。实现打开、关闭设备、配置、报文收发等功能。接口库采用Visual Studio 2022开发，依赖运行库2022版本，需要确保计算机中包含该运行库，否则需从微软官网下在安装。该函数库支持我司PCIe、MINI PCIe接口的CAN、CANFD系列的接口卡。

开发编程时直接加载CANFD.dll即可，接口描述文件位于CANFD.h，dll文件需要位于可执行文件的同级目录下即可。

## 2.2 库函数调用流程

# 三、接口卡参数

## 3.1 型号参数

|  |  |
| --- | --- |
| **产品型号** | **类型号** |
| SG-PCIe-CAN-200T | 1 |
| SG-PCIe-CANFD-200T | 2 |
| SG-MINI PCIe-CAN-200T | 3 |
| SG-MINI PCIe-CANFD-200T | 4 |

## 3.2 库函数数据结构说明

### 3.2.1 CAN初始化数据类型

typedef struct {

BYTE FDFlag;

BYTE NON\_ISO;

BYTE Timing0;

BYTE Timing1;

BYTE Mode;

BYTE reserved[3];

}VCI\_INIT\_CONFIG, \*PVCI\_VCI\_INIT\_CONFIG;

FDFlag CAN协议类型，0：CAN2.0 1：CANFD。

NON\_ISO CAN标准， 0：ISO标准 1：NON\_ISO标准。

Timing0 CAN仲裁域波特率（见下表）。

Timing1 CAN 数据域波特率（见下表）。

Mode CAN 工作模式，0：正常模式 1：只听模式 2：自发自收模式。

reserved[3] 保留。

|  |  |  |  |
| --- | --- | --- | --- |
| **仲裁域波特率** | **Timing0** | **数据域波特率** | **Timing1** |
| 5Kbps | 0x10 | 100Kbps | 0x10 |
| 10Kbps | 0x20 | 125Kbps | 0x20 |
| 20Kbps | 0x30 | 200Kbps | 0x30 |
| 25Kbps | 0x40 | 250Kbps | 0x40 |
| 50Kbps | 0x50 | 500Kbps | 0x50 |
| 100Kbps | 0x60 | 800Kbps | 0x60 |
| 125Kbps | 0x70 | 1000Kbps | 0x70 |
| 250Kbps | 0x80 | 1500Kbps | 0x80 |
| 500Kbps | 0x90 | 2000Kbps | 0x90 |
| 800Kbps | 0xA0 | 3000Kbps | 0xA0 |
| 1000Kbps | 0xB0 | 5000Kbps | 0xB0 |

### 3.2.2 CAN信息帧数据类型

typedef struct {

DWORD ID;

BYTE RemoteFlag;

BYTE ExternFlag;

BYTE FDFlag;

BYTE DataLen;

BYTE Date[64];

} VCI\_CAN\_OBJ, \* PVCI\_CAN\_OBJ;

ID CAN ID。

RemoteFlag 帧类型， 0：数据帧 1：远程帧。

ExternFlag 帧格式， 0：标准帧 1：扩展帧。

FDFlag CAN类型 0：CAN2.0 1：CANFD。

DataLen 数据长度，CAN的数据长度取值为：0 ~8，CANFD的数据长度 取值为： 0 ~8，12，16，20，24，32，48，64 字节。

Data CAN数据。

### 3.2.3 CAN 滤波器数据类型

typedef struct {

DWORD FilterIDEnable;

DWORD FilterIDCount;

DWORD FilterIDType[5];

DWORD FilterIDMin[5];

DWORD FilterIDMax[5];

}VCI\_FILTER\_CONFIG,\*P\_VCI\_FILTER\_CONFIG; can记录仪 说明书.docx

**FilterIDEnable** 滤波使能，0：不使能，1：使能。

**FilterIDCount** 滤波ID的数目，取值范围为：0~5。

**FilterIDType** 滤波ID类型，0：标准帧，1：扩展帧。

**FilterIDMin** 滤波ID的最小值。

**FilterIDMax** 滤波ID的最大值。

## 3.3 接口函数说明

**[1] 打开设备**

DWORD \_\_stdcall VCI\_OpenDevice(DWORD DevType, DWORD DevIndex);

**DevType** 设备类型号，对应不同的产品型号，见表2-1。

**DevIndex** 设备索引号，有一个设备时索引号为0，有两个可以为0或1。

**返回值** 为1 表示操作成功，0 表示操作失败。

**[2] 关闭设备**

DWORD \_\_stdcall VCI\_CloseDevice(DWORD DevType, DWORD DevIndex);

**DevType** 设备类型号。

**DevIndex** 设备索引号，有一个设备时索引号为0，有两个可以为0或1。

**返回值** 为1 表示操作成功，0 表示操作失败。

**[3] 初始化CAN**

DWORD \_stdcall VCI\_InitCan(DWORD DevType, DWORD DevIndex, DWORD

CANIndex, P\_VCI\_INIT\_CONFIG InitConfig);

**DevType** 设备类型号。

**DevIndex** 设备索引号，有一个设备时索引号为0，有两个可以为0或1。

**CANIndex**第几路CAN。

**InitConfig** 初始化参数结构体。

**返回值** 为1 表示操作成功，0 表示操作失败。

**[4] 发送数据**

DWORD \_\_stdcall VCI\_Transmit(DWORD DevType, DWORD DevIndex，

DWORD CANIndex, P\_VCI\_CAN\_OBJ \*pSend, DWORD Len);

**DevType** 设备类型号。

**DevIndex** 设备索引号，有一个设备时索引号为0，有两个可以为0或1。

**CANIndex**第几路CAN。

**pSend** 要发送的信息帧结构体数组首地址，介绍参见函数库数据结构部分。

**Len** 要发送的帧结构体数组的长度（发送的帧数量）。

**返回值** 返回实际发送成功的帧数。

**[5] 接受数据**

DWORD \_\_stdcall VCI\_Receive(DWORD DevType, DWORD DevIndex ，

DWORD CANIndex，PVCI\_CAN\_OBJ pReceive , DWORD Len , DWORD WaitTime);

**DevType** 设备类型号。

**DevIndex** 设备索引号，有一个设备时索引号为0，有两个可以为0或1。

**pReceive** 用来接收的数据帧结构体数组的首指针。

**Len** 读取多少帧的数据。

**WaitTime** 以毫秒为单位。

**返回值** 返回实际读取到的帧数。如果返回值为0，则表示没有读到数据。

**[6] 获取缓冲区中尚未读取的帧数**

DWORD \_\_stdcall VCI\_GetReceiveNum(DWORD DevType, DWORD DevIndex,

DWORD CANIndex);

**DevType** 设备类型号。

**DevIndex** 设备索引号，有一个设备时索引号为0，有两个可以为0或1。

**CANIndex** 第几路CAN。

**返回值** 返回缓冲区中尚未读取的帧数。

**[7] 清空缓冲区中的数据**

BOOL \_\_stdcall VCI\_ClearBuffer(DWORD DevType, DWORD DevIndex,DWORD

CANIndex);

**DevType** 设备类型号。

**DevIndex** 设备索引号，有一个设备时索引号为0，有两个可以为0或1。

**CANIndex** 第几路CAN。

**返回值** 为1 表示操作成功，0 表示操作失败。

**[8] 滤波设置**

DWORD \_\_stdcall VCI\_SetValue(DWORD DevType, DWORD DevIndex, DWORD

CANIndex, P\_VCI\_FILTER\_CONFIG pInitConfig);

**DevType** 设备类型号。

**DevIndex** 设备索引号，有一个设备时索引号为0，有两个可以为0或1。

**CANIndex** 第几路CAN。

**pInitConfig** 滤波器结构体。

**返回值** 为1 表示操作成功，0 表示操作失败。

# 四、接口库使用方法

首先，把库函数文件都放在工作目录下。库函数文件有三个：CANFD.h、CANFD.lib、CANFD.dll。

## 4.1 VC调用动态库方法

(1) 在扩展名为.CPP 的文件中包含CANFD.h 头文件。

如：#include“CANFD.h”。

(2) 在工程的连接器设置中连接到CANFD.lib 文件。

如：在VS 环境下，在项目属性页里的配置属性-->连接器-->输入-->附加依赖项 中 添加CANFD.lib。

# **五、售后及联系方式**

公司网址： [www.tj-sange.com](http://www.tj-sange.com)

售后联系电话：022-22106681  13072208083（微信）

**公众账号：获取产品使用视频和更多资讯。**