PCHCAN115 WIN2000/XP 驱动程序使用说明书

请您务必阅读《使用纲要》，他会使您事半功倍!

目　　录

PCHCAN115 WIN2000/XP 驱动程序使用说明书.....1
第一章　版权信息与命名约定.....2
第一节、版权信息.....2
第二节、命名约定.....2
第二章　PCHCAN 设备专用函数接口介绍.....2
第一节、设备驱动接口函数列表（每个函数省略了前缀“PCHCAN115_”）.....2
第二节、设备对象管理函数原型说明.....4
第三节、寄存器操作函数原型说明.....9
第三章　硬件参数结构.....10
第一节、 CAN 设备初始化结构介绍（CAN_INIT_PARA_）.....10
第二节、CAN 帧结构（CAN_FRAME_）.....11

提醒用户：

通常情况下，WINDOWS 系统在安装时自带的 DLL 库和驱动不全，所以您不管使用那种语言编程，请您最好先安装上Visual C++6.0 版本的软件，方可使我们的驱动程序有更完备的运行环境。

有关设备驱动安装和产品二次发行请参考 PCHCAN115Inst.doc 文档。

第一章 版权信息与命名约定

第一节、版权信息

第二节、命名约定

一、为简化文字内容，突出重点，本文中提到的函数名通常为基本功能名部分，其前缀设备名如PCHCANxxxx_则被省略。如PCHCAN115_CreateDevice 则写为CreateDevice。二、函数名及参数中各种关键字缩写规则

缩写	全称	汉语意思	缩写	全称	汉语意思
Dev	Device	设备	DI	Digital Input	数字量输入
Pro	Program	程序	DO	Digital Output	数字量输出
Int	Interrupt	中断	CNT	Counter	计数器
Dma	Direct Memory Access	直接内存存取	DA	Digital convert to Analog	数模转换
AD	Analog convert to Digital	模数转换	DI	Differential	(双端或差分) 注：在常量选项中
Npt	Not Empty	非空	SE	Single end	单端
Para	Parameter	参数	DIR	Direction	方向
SRC	Source	源	ATR	Analog Trigger	模拟量触发
TRIG	Trigger	触发	DTR	Digital Trigger	数字量触发
CLK	Clock	时钟	Cur	Current	当前的
GND	Ground	地	OPT	Operate	操作
Lgc	Logical	逻辑的	ID	Identifier	标识
Phys	Physical	物理的			

以上规则不局限于该产品。

第二章 PCHCAN 设备专用函数接口介绍

第一节、设备驱动接口函数列表（每个函数省略了前缀“PCHCAN115_”）

函数名	函数功能	备注
① 设备对象操作函数		
CreateDevice	创建设备对象	
ReleaseDevice	释放设备对象	
ResetDevice	复位 CAN 卡为出厂状态	
GetDeviceCurrentID	取得 CAN 卡设备逻辑ID和物理ID	
ListDeviceDlg	列表设备	
InitCAN	初始化 CAN 卡	
SendFrame	发送单帧操作	

ReceiveFrame	接收帧操作	
② 寄存器操作		
DEV_Write	写CAN寄存器数据	
DEV_Read	读CAN寄存器数据	

使用需知

Visual C++ & C++Builder&LabWindow：

首先将 PCHCAN115.h 和 PCHCAN115.lib 两个驱动库文件从相应的演示程序文件夹下复制到您的源程序文件夹中，然后在您的源程序头部添加如下语句，以便将驱动库函数接口的原型定义信息和驱动接口导入库(PCHCAN115.lib)加入到您的工程中。

#include “PCHCAN115.H”

在 VC 中，为了使用方便，避免重复定义和包含，您最好将以上语句放在StdAfx.h 文件。一旦完成了以上工作，那么使用设备的驱动程序接口就跟使用VC/C++Builder/LabWindow 自身的各种函数，其方法一样简单，毫无二别。

关于PCHCAN115.h 和 PCHCAN115.lib 两个文件均可在演示程序文件夹下面找到。

Visual Basic:

首先将PCHCAN115.Bas 驱动模块头文件从 VB 的演示程序文件夹下复制到您的源程序文件夹中，然后将此模块文件加入到您的 VB 工程中。其方法是选择 VB 编程环境中的工程(Project)菜单，执行其中的"添加模块"(Add Module)命令,在弹出的对话中选择PCHCAN115.Bas 模块文件即可，一旦完成以上工作后，那么使用设备的驱动程序接口就跟使用 VB 自身的各种函数，其方法一样简单，毫无二别。

请注意，因考虑Visual C++和 Visual Basic 两种语言的兼容问题，在下列函数说明和示范程序中，所举的Visual Basic 程序均是需要编译后在独立环境中运行。所以用户若在解释环境中运行这些代码，我们不保证能完全顺利运行。

Delphi:

首先将 PCHCAN115.Pas 驱动模块头文件从 Delphi 的演示程序文件夹下复制到您的源程序文件夹中， 然后将此模块文件加入到您的 Delphi 工程中。其方法是选择 Delphi 编程环境中的 View 菜单,执行其中的"Project Manager"命令,在弹出的对话中选择*.exe 项目，再单击鼠标右键，最后 Add 指令，即可将PCHCAN115.Pas 单元模块文件加入到工程中。或者在Delphi 的编程环境中的Project 菜单中，执行 Add To Project 命令，然后选择*.Pas 文件类型也能实现单元模块文件的添加。最后请在使用驱动程序接口的源程序文件中的头部的Uses 关键字后面的项目中加入：“PCHCAN115”。如：

```
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
PCHCAN115; // 注意： 在此加入驱动程序接口单元PCHCAN115
```

LabView / CVI:

LabVIEW 是美国国家仪器公司(National Instrument)推出的一种基于图形开发、调试和运行程序的集成化环境，是目前国际上唯一的编译型的图形化编程语言。在以 PC 机为基础的测量和工控软件中，LabVIEW 的市场普及率仅次于C++/C 语言。LabVIEW 开发环境具有一系列优点，从其流程图式的编程、

不需预先编译就存在的语法检查、调试过程使用的数据探针，到其丰富的函数功能、数值分析、信号处理和设备驱动等功能，都令人称道。关于LabView/CVI 的驱动程序接口的详细说明请参考其演示源程序。

第二节、设备对象管理函数原型说明

♦ 创建设备对象函数

函数原型：

Visual C++ & C++ Builder&LabWindow：

HANDLE CreateDevice(int DeviceID)

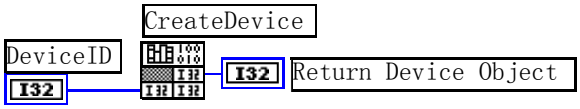
Visual Basic：

Declare Function CreateDevice Lib "PCHCAN115" (ByVal DeviceID As Long) As Long

Delphi：

Function CreateDevice(DeviceID:LongInt):LongInt; StdCall; External 'PCHCAN115' Name 'CreateDevice';

LabView:



功能：该函数负责创建设备对象，并返回其设备对象句柄。

参数：

DeviceID设备逻辑ID(Identifier)标识号。当向同一个Windows 系统中加入若干相同类型的设备时，系统将以该设备的“基本名称”与 ID 标识值为名称后缀的标识符来确认和管理该设备。比如若用户往 Windows 系统中加入第一个PCHCAN115 AD 模板时，系统则以“PCHCAN115”作为基本名称，再以 ID 的初值组合成该设备的标识符“PCHCAN115-0”来确认和管理这第一个设备，若用户接着再添加第二个 PCHCAN115 AD 模板时，则系统将以“PCHCAN115-1”来确认和管理第二个设备，若再添加，则以此类推。所以当用户要创建设备句柄管理和操作第一个 CAN 设备时，ID 应置 0，第二应置 1，也以此类推。默认值为 0。

返回值：如果执行成功，则返回设备对象句柄；如果没有成功，则返回错误码 INVALID_HANDLE_VALUE。由于此函数已带容错处理，即若出错，它会自动弹出一个对话框告诉您出错的原因。您只需要对此函数的返回值作一个条件处理即可，别的任何事情您都不必做。

备注：创建完成后，如果释放 CAN 需要用PCHCAN115_ReleaseDevice 来关闭 CAN卡。

相关函数：ReleaseDevice

♦ 释放设备对象所占的系统资源及设备对象

函数原型：

Visual C++ & C++Builder&LabWindow:

BOOL ReleaseDevice(HANDLE hDevice)

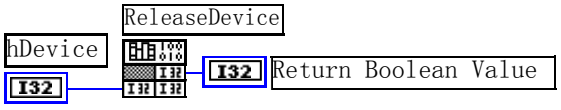
Visual Basic:

Declare Function ReleaseDevice Lib "PCHCAN115" (ByVal hDevice As Long) As Boolean

Delphi:

Function ReleaseDevice(hDevice : LongInt):Boolean; StdCall; External 'PCHCAN115' Name 'ReleaseDevice';

LabView:



功能：释放设备对象所占用的系统资源及设备对象自身

参数：hDevice 设备对象句柄，它应由CreateDevice创建。

返回值：若成功，则返回TRUE, 否则返回 FALSE。

相关函数：CreateDevice

应注意的是，CreateDevice必须和ReleaseDevice函数一一对应，即当您执行了一次CreateDevice，再一次执行这个函数前，必须执行一次ReleaseDevice函数，以释放由CreateDevice占用的系统软硬件资源，如系统内存等。只有这样，当您再次调用CreateDevice函数时，那些软硬件资源才可被再次使用。

- **复位 PCHCAN**

卡函数原型：

Visual C++ & C++Builder&LabWindow:

```
BOOL ResetDevice(HANDLE  hDevice,LONG lChannel);
```

Visual Basic:

```
Declare Function ResetDevice Lib "PCHCAN115" (ByVal hDevice As Long,ByVal lChannel As Long ) As Boolean
```

Delphi:

```
Function ResetDevice (hDevice : LongInt,lChannel :LongInt):Boolean; StdCall; External 'PCHCAN115' Name 'ResetDevice';
```

LabView:

请参考相关演示程序。

功能：复位整个 PCHCAN 设备，相当于它与 PC 机端重新建立。其效果与重新插上 PCHCAN 设备等同。一般在出错情况下，想软复位来建决重连接问题，就可以调用该函数解决此问题。

参数：hDevice 设备对象句柄，它应由CreateDevice创建。由它指向要复位的设备。

lChannel CAN通道[0~1]

返回值：若成功，则返回 TRUE, 否则返回 FALSE。

相关函数：CreateDevice ReleaseDevice

- **取得当前设备对象句柄指向的设备所在的设备 ID**

函数原型：

Visual C++ & C++Builder&LabWindow:

```
BOOL GetDeviceCurrentID(HANDLE hDevice, PLONG DeviceLgcID, PLONG DevicePhysID)
```

Visual Basic:

```
Declare Function GetDeviceCurrentID Lib "PCHCAN115" (ByVal hDevice As Long, _
                                                     ByRef DeviceLgcID As Long,_
                                                     ByRef DevicePhysID As Long) As Boolean
```

Delphi:

```
Function GetDeviceCurrentID(hDevice : LongInt;
                            DeviceLgcID : PLong;
                            DevicePhysID :PLong): Boolean;
                            StdCall; External 'PCHCAN115' Name ' GetDeviceCurrentID';
```

LabView:

请参考相关演示程序。

功能：取得指定设备对象所代表的设备在设备链中的物理设备 ID 号和逻辑 ID 号。

参数：

hDevice 设备对象句柄，它应由CreateDevice创建。
DeviceLgcID 返回设备逻辑ID。
DevicePhysID 返回设备物理ID。
返回值：若成功，则返回 TRUE，否则返回 FALSE。
相关函数：CreateDevice ReleaseDevice

♦ **列表计算机系统中的所有的该PCH设备**

函数原型：

Visual C++ & C++Builder&LabWindow:
BOOL ListDeviceDlg(HANDLE hDevice)
Visual Basic:
Declare Function ListDeviceDlg Lib "PCHCAN115" (ByVal hDevice As Long) As Boolean
Delphi:
Function ListDeviceDlg (hDevice : LongInt): Boolean;StdCall; External 'PCHCAN115' Name ' ListDeviceDlg' ;
LabView:
请参考相关演示程序。

功能：列表系统当中的所有的该PCH设备。
参数：
hDevice 设备对象句柄，它应由CreateDevice创建。
返回值：若成功，则返回 TRUE，否则返回 FALSE。
相关函数：CreateDevice ReleaseDevice

♦ **取得设备总台数**

函数原型：

Visual C++ & C++Builder&LabWindow:
Long GetDeviceCount(HANDLE hDevice)
Visual Basic:
Declare Function GetDeviceCount Lib "PCHCAN115" (ByVal hDevice As Long) As Long
Delphi:
Function GetDeviceCount (hDevice : LongInt): LongInt;StdCall; External 'PCHCAN115' Name ' GetDeviceCount' ;
LabView:
请参考相关演示程序。

功能：取得设备总台数。
参数：
hDevice 设备对象句柄，它应由CreateDevice创建。
返回值：返回设备的数量
相关函数：CreateDevice ReleaseDevice

♦ **初始化设备对象**

函数原型：

Visual C++ & C++Builder&LabWindow:
BOOL InitCAN(HANDLE hDevice,
PPCHCAN115_INIT_PARA pPara,

```
                    LONG lChannel)
```

Visual Basic:

```
Declare Function InitCAN Lib "PCHCAN115" (ByVal hDevice As Long, _
                                          ByRef pPara As PCHCAN115_INIT_PARA, _
                                          ByVal lChannel as Long) As Boolean
```

Delphi:

```
Function InitCAN (hDevice : LongInt;
                  pPara:PPCHCAN115_INIT_PARA;
                  lChannel:LongInt): Boolean;StdCall; External 'PCHCAN115' Name ' InitCAN';
```

LabView:

请参考相关演示程序。

功能：初始化CAN卡，进入工作模式。

参数：

hDevice 设备对象句柄，它应由CreateDevice创建。

pPara 初始化参数

lChannel CAN通道[0~1]

返回值：若成功，则返回 TRUE，否则返回 FALSE。

相关函数：CreateDevice ReleaseDevice

♦ **发送单帧操作**

函数原型：

Visual C++ & C++Builder&LabWindow:

```
Long SendFrame(HANDLE hDevice,
               PPCHCAN115_FRAME psCanFrame,
               PLONG pSts,
               double dTimeOut,
               LONG lChannel);
```

Visual Basic:

```
Declare Function SendFrame Lib "PCHCAN115" (ByVal hDevice As Long, _
                                            ByRef psCanFrame As PCHCAN115_FRAME, _
                                            ByRef pSts As Long, _
                                            ByVal dTimeOut As Double, _
                                            ByVal lChannel as Long) As Long
```

Delphi:

```
Function SendFrame (hDevice : LongInt;
                    psCanFrame : PPCHCAN115_FRAME;
                    pSts : PLong;
                    dTimeOut :Double;
                    lChannel:LongInt): LongInt;StdCall; External 'PCHCAN115' Name 'SendFrame';
```

LabView:

请参考相关演示程序。

功能：发送单帧操作。

参数：

hDevice 设备对象句柄，它应由CreateDevice创建。
psCanFrame CAN帧
pSts 返回状态信息
dTimeOut 发送超时单位ms
lChannel CAN通道[0~1]
返回值：如果成功：发送字节数 如果失败：0。
相关函数：CreateDevice ReleaseDevice

♦ **接收帧操作**

函数原型：

Visual C++ & C++Builder&LabWindow:

```
Long ReceiveFrame(HANDLE hDevice,
                  PPCHCAN115_FRAME pCanFrame,
                  PLONG pSts,
                  double dTimeOut,
                  LONG lChannel);
```

Visual Basic:

```
Declare Function ReceiveFrame Lib "PCHCAN115" (ByVal hDevice As Long, _
                                               ByRef pCanFrame As PCHCAN115_FRAME, _
                                               ByRef pSts As Long, _
                                               ByVal dTimeOut As Double, _
                                               ByVal lChannel as Long) As Long
```

Delphi:

```
Function ReceiveFrame (hDevice : LongInt;
                       pCanFrame : PPCHCAN115_FRAME;
                       pSts : PLong;
                       dTimeOut :Double;
                       lChannel:LongInt): LongInt;StdCall; External 'PCHCAN115' Name 'ReceiveFrame';
```

LabView:

请参考相关演示程序。

功能：发送单帧操作。
参数：
hDevice 设备对象句柄，它应由CreateDevice创建。
pCanFrame CAN帧
pSts 返回状态信息
dTimeOut 发送超时单位ms
lChannel CAN通道[0~1]
返回值：如果成功：接收字节数 如果失败：0。
相关函数：CreateDevice ReleaseDevice

第三节、寄存器操作函数原型说明

♦ 写CAN寄存器数据

函数原型：

Visual C++ & C++Builder&LabWindow:

```
BOOL DEV_Write(HANDLE hDevice,
               BYTE bAddr,
               BYTE bData,
               LONG lChannel);
```

Visual Basic:

```
Declare Function DEV_Write Lib "PCHCAN115" (ByVal hDevice As Long, _
                                            ByVal bAddr As Byte, _
                                            ByVal bData As Byte, _
                                            ByVal lChannel as Long) As Boolean
```

Delphi:

```
Function DEV_Write (hDevice : LongInt;
                    bAddr : Byte;
                    bData : Byte;
                    lChannel:LongInt): Boolean;StdCall; External 'PCHCAN115' Name 'DEV_Write';
```

LabView:

请参考相关演示程序。

功能：写CAN寄存器数据。

参数：

hDevice 设备对象句柄，它应由CreateDevice创建。

bAddr 地址

bData 数据

lChannel CAN通道[0~1]

返回值：若成功，则返回 TRUE，否则返回 FALSE。

相关函数：CreateDevice　　ReleaseDevice

♦ 读CAN寄存器数据

函数原型：

Visual C++ & C++Builder&LabWindow:

```
BYTE DEV_Read(HANDLE hDevice,
              BYTE bAddr,
              LONG lChannel);
```

Visual Basic:

```
Declare Function DEV_Read Lib "PCHCAN115" (ByVal hDevice As Long, _
                                           ByVal bAddr As Byte, _
                                           ByVal lChannel as Long) As Byte
```

Delphi:

```
Function  DEV_Read (hDevice : LongInt;
                    bAddr : Byte;
                    lChannel:LongInt): Byte;StdCall; External 'PCHCAN115'  DEV_Read';
```

LabView:

请参考相关演示程序。

功能：写CAN寄存器数据。

参数：

hDevice 设备对象句柄，它应由CreateDevice创建。

bAddr 地址

lChannel CAN通道[0~1]

返回值：若成功，则返回读到的数据。

相关函数：CreateDevice ReleaseDevice

第三章 硬件参数结构

第一节、CAN 设备初始化结构介绍（CAN_INIT_PARA_）

Visual C++ & C++Builder&LabWindow:

```
typedef  struct _INIT_PARA_
{
        ULONG       WorkMode;           // 工作方式，=0时正常模式，=1时为只听模式。....
        ULONG       AFilterMode;    // 滤波模式，=0双滤波器模式，=1单滤波器模式.
        ULONG       SFType;                 // 帧类型发送帧类型，=1时为自发自收
        ULONG       CheckCode;          // 验收码
        ULONG       MaskOffCode;   // 屏蔽码
        ULONG       BaudRate;           // 总线速率, 当值为x09(自定义)时，定时器定时器起作用
        ULONG       Timer0;             // 定时器
        ULONG       Timer1;             // 定时器
} _INIT_PARA, *_INIT_PARA;
```

Visual Basic:

```
Type  _INIT_PARA_
   ByVal WorkMode As Long     ‘工作方式，=0时正常模式，=1时为只听模式。
   ByVal AFilterMode As Long   ‘ 滤波模式，=0双滤波器模式，=1单滤波器模式
   ByVal SFType As Long              ‘ 帧类型发送帧类型，=1时为自发自收
   ByVal CheckCode As Long     ‘ 验收码
   ByVal MaskOffCode As Long ‘ 屏蔽码
   ByVal BaudRate As Long       ‘ 总线速率, 当值为x09(自定义)时，定时器定时器起作用
   ByVal Timer0 As Long              ‘ 定时器
   ByVal Timer1As Long               ‘ 定时器
End Type
```

Delphi:

```
Type
  _PINIT_PARA_ = ^ _INIT_PARA_; // 指针类型结构
  _INIT_PARA_ = record // 标记为记录型

   WorkMode : LongWord;        // 工作方式，=0时正常模式，=1时为只听模式。
   AFilterMode : LongWord;      // 滤波模式，=0双滤波器模式，=1单滤波器模式
   SFType : LongWord;          // 帧类型发送帧类型，=1时为自发自收
   CheckCode : LongWord;       // 验收码
   MaskOffCode : LongWord;     // 屏蔽码
   BaudRate : LongWord;        // 总线速率, 当值为x09(自定义)时，定时器定时器起作用
   Timer0 : LongWord;          // 定时器
   Timer1 : LongWord;          // 定时器
End
```

LabView:
请参考其演示程序。

硬件参数说明：此结构主要用于设定设备硬件参数值，用这个参数结构对设备进行硬件配置由InitCAN函数完成。
WorkMode工作方式选择。它的其选项值如下表：

常量名	常量值	功能定义
PCHCAN115_WORKMODE_NORMAL	0X00	正常模式
PCHCAN115_WORKMODE_LISTEN	0X01	只听模式
PCHCAN115_WORKMODE_SELFTEST	0X02	自检测模式

AFilterMode工作方式。它的其选项值如下表：

常量名	常量值	功能定义
PCHCAN115_AFM_DOUBLE	0X00	双滤波器模式
PCHCAN115_AFM_SINGLE	0X01	单滤波器模式

SFType所使用发送帧类型：

常量名	常量值	功能定义
PCHCAN115_AFM_DOUBLE	0X00	正常发送
PCHCAN115_AFM_SINGLE	0X01	自发自收

BaudRate所使用波特率：

常量名	常量值	功能定义
PCHCAN115_BAUD_10KHZ	0X00	10Kbps
PCHCAN115_BAUD_20KHZ	0X01	20Kbps
PCHCAN115_BAUD_50KHZ	0X02	50Kbps
PCHCAN115_BAUD_100KHZ	0X03	100Kbps
PCHCAN115_BAUD_125KHZ	0X04	125Kbps
PCHCAN115_BAUD_250KHZ	0X05	250Kbps
PCHCAN115_BAUD_500KHZ	0X06	500Kbps
PCHCAN115_BAUD_800KHZ	0X07	800Kbps
PCHCAN115_BAUD_1MHZ	0X08	1Mbps
PCHCAN115_BAUD_CUSTOM	0X09	自定义

第二节、CAN 帧结构（CAN_FRAME_）

Visual C++ & C++Builder&LabWindow:

```
typedef struct _FRAME_
{
   ULONG      FrameFormat;   // 帧格式          =0为数据帧，=1为远程帧
   ULONG      FrameType;              // 帧类型          =0为标准帧，=1为扩展帧
   ULONG      DataLen;                // 数据长度<=8
   BYTE       Data[8];          // 数据
   ULONG      dwID;                   // 设备ID
   ULONG      dwSigntime;             // 接收帧时间标识
} PCHCAN115_FRAME, *PPCHCAN115_FRAME;
```

Visual Basic:

```
Type _FRAME
```

```
    ByVal FrameFormat As Long  '帧格式=0为数据帧，=1为远程帧
    ByVal FrameType As Long     ' 帧类型        =0为标准帧，=1为扩展帧
    ByVal DataLen As Long                ' 数据长度<=8
    ByVal Data(0 to 7) As Byte    ' 数据
    ByVal dwID As Long ' 设备ID
    ByVal dwSigntime As Long    ' 接收帧时间标识
  End Type
```

Delphi:

```
Type
  _PFRAME = ^ _FRAME; // 指针类型结构
  _FRAME = record // 标记为记录型

        FrameFormat : LongWord;       // 帧格式        =0为数据帧，=1为远程帧
        FrameType : LongWord;         // 帧类型        =0为标准帧，=1为扩展帧
        DataLen : LongWord;           // 数据长度<=8
        Data : Array[0..15]  of  BYTE;    // 数据
        dwID : LongWord;                    // 设备ID
        dwSigntime : LongWord;        // 接收帧时间标识
End
```

LabView:

请参考其演示程序。

各参数说明：

FrameFormat所使用帧格式。它的其选项值如下表：

常量名	常量值	功能定义
PCHCAN115_FFORMAT_DATA	0X00	数据帧
PCHCAN115_FFORMAT_REMOTE	0X01	远程帧

FrameType所使用帧类型：

常量名	常量值	功能定义
PCHCAN115_FTYPE_STANDARD	0X00	标准帧
PCHCAN115_FTYPE_EXTEND	0X01	扩展帧