PCH2542 数据采集卡

WIN2000/XP 驱动程序使用说明书

阿尔泰科技发展有限公司

产品研发部修订

请您务必阅读《使用纲要》，他会使您事半功倍!

目　录

目　录 1
第一章 版权信息与命名约定 1
第一节、版权信息 1
第二节、命名约定 1
第二章 使用纲要 2
第一节、哪些函数对您不是必须的 2
第三章 PCH设备操作函数接口介绍 2
第一节、设备驱动接口函数总列表（每个函数省略了前缀“PCH2542_”） 3
第二节、设备对象管理函数原型说明 4
第二节、计数器控制函数原型说明 6
第四章 上层用户函数接口应用实例 8
第一节、简易程序演示说明 8
第二节、高级程序演示说明 9
第五章 共用函数介绍 9
第一节、公用接口函数总列表（每个函数省略了前缀“PCH2542_”） 9
第二节、PCH内存映射寄存器操作函数原型说明 10
第三节、IO端口读写函数原型说明 18
第四节、线程操作函数原型说明 21
第五节、文件对象操作函数原型说明 21
第六节、各种参数保存和读取函数原型说明 24
第七节、其他函数原型说明 26

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 PCHxxxx_则被省略。如 PCH2542_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	模数转换	DI	Differential	(双端或差分) 注：

	to Digital				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 使用纲要

第一节、哪些函数对您不是必须的

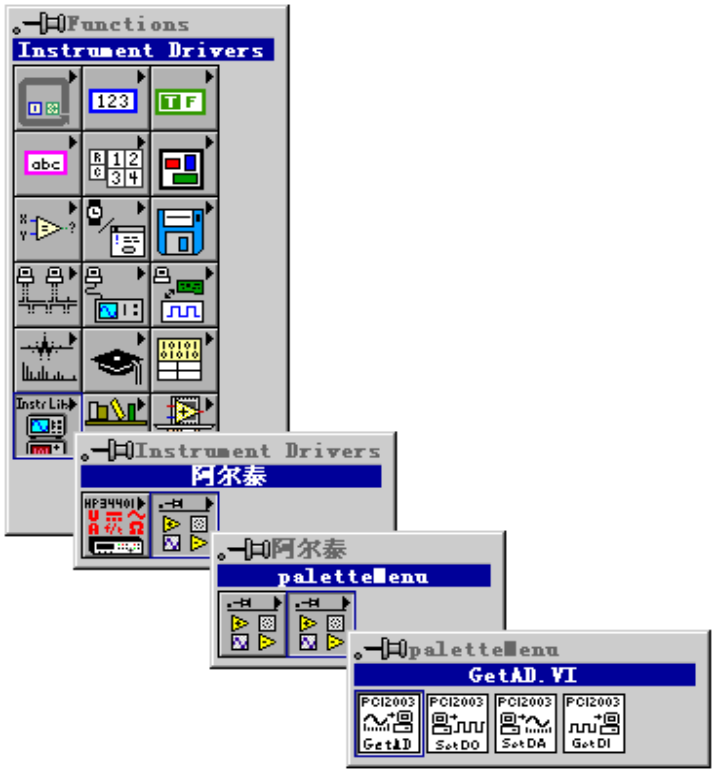
公共函数如CreateFileObject，WriteFile，ReadFile等一般来说都是辅助性函数，除非您要使用存盘功能。如果您使用上层用户函数访问设备，那么GetDeviceAddr，WriteRegisterByte，WriteRegisterWord，WriteRegisterULong，ReadRegisterByte，ReadRegisterWord，ReadRegisterULong等函数您可完全不必理会，除非您是作为底层用户管理设备。而WritePortByte，WritePortWord，WritePortULong，ReadPortByte，ReadPortWord，ReadPortULong则对PCH用户来讲，可以说完全是辅助性，它们只是对我公司驱动程序的一种功能补充，对用户额外提供的，它们可以帮助您在NT、Win2000 等操作系统中实现对您原有传统设备如ISA卡、串口卡、并口卡的访问，而没有这些函数，您可能在基于Windows NT架构的操作系统中无法继续使用您原有的老设备。

第三章 PCH 设备操作函数接口介绍

由于我公司的设备应用于各种不同的领域，有些用户可能根本不关心硬件设备的控制细节，只关心AD的首末通道、采样频率等，然后就能通过一两个简易的采集函数便能轻松得到所需要的AD数据。这方面的用户我们称之为上层用户。那么还有一部分用户不仅对硬件控制熟悉，而且由于应用对象的特殊要求，则要直接控制设备的每一个端口，这是一种复杂的工作，但又是必须的工作，我们则把这一群用户称之为底层用户。因此总的看来，上层用户要求简单、快捷，他们最希望在软件操作上所要面对的全是他们最关心的问题，比如在正式采集数据之前，只须用户调用一个简易的初始化函数（如InitDeviceProAD）告诉设备我要使用多少个通道，采样频率是多少赫兹等，然后便可以用ReadDeviceProAD Npt（或ReadDeviceProAD Half）函数指定每次采集的点数，即可实现数据连续不间断采样。而关于设备的物理地址、端口分配及功能定义等复杂的硬件信息则与上层用户无任何关系。那么对于底层用户则不然。他们不仅要关心设备的物理地址，还要关心虚拟地址、端口寄存器的功能分配，甚至每个端口的Bit位都要了如指掌，看起来这是一项相当复杂、繁琐的工作。但是这些底层用户一旦使用我们提供的技术支持，则不仅可以让您不必熟悉PCH总线复杂的控制协议，同是还可以省掉您许多繁琐的工作，比如您不用去了解PCH的资源配置空间、PNP即插即用管理，而只须用GetDeviceAddr函数便可以同时取得指定设备的物理基地址和虚拟线性基地址。这个时候您便可以用这个虚拟线性基地址，再根据硬件使用说明书中的各端口寄存器的功能说明，然后使用ReadRegisterULong和WriteRegisterULong对这些端口寄存器进行 32 位模式的读写操作，即可实现设备的所有控制。

综上所述，用户使用我公司提供的驱动程序软件包将极大的方便和满足您的各种需求。但为了您更省心，别忘了在您正式阅读下面的函数说明时，先明白自己是上层用户还是底层用户，因为在《设备驱动接口函数总列表》中的备注栏里明确注明了适用对象。

另外需要申明的是，在本章和下一章中列明的关于 LabView 的接口，均属于外挂式驱动接口，他是通过 LabView 的 Call Labrary Function 功能模板实现的。它的特点是除了自身的语法略有不同以外，每一个基于 LabView 的驱动图标与 Visual C++、Visual Basic、Delphi 等语言中每个驱动函数是一一对应的，其调用流程和功能是完全相同的。那么相对于外挂式驱动接口的另一种方式是内嵌式驱动。这种驱动是完全作为 LabView 编程环境中的紧密耦合的一部分，它可以直接从 LabView 的 Functions 模板中取得，如下图所示。此种方式更适合上层用户的需要，它的最大特点是方便、快捷、简单，而且可以取得它的在线帮助。关于 LabView 的外挂式驱动和内嵌式驱动更详细的叙述，请参考 LabView 的相关演示。



LabView 内嵌式驱动接口的获取方法

第一节、设备驱动接口函数总列表（每个函数省略了前缀“PCH2542_”）

函数名	函数功能	备注
① 设备对象操作函数		
CreateDevice	创建 PCH 设备对象(用设备逻辑号)	上层及底层用户
GetDeviceCount	取得同一种 PCH 设备的总台数	上层及底层用户
GetDeviceCurrentID	取得指定设备的逻辑 ID 和物理 ID	上层及底层用户
ListDeviceDlg	列表所有同一种 PCH 设备的各种配置	上层及底层用户
ReleaseDevice	关闭设备，且释放 PCH 总线设备对象	上层及底层用户
②计数器控制函数		
StartDeviceCNT	启动设备计数	上层用户
SetDeviceCNT	设置计数器的初值	上层用户
GetDeviceCNT	取得各路计数器的当前计数值	上层用户
StopDeviceCNT	停止设备计数	上层用户

使用需知：

Visual C++：

要使用如下函数关键的问题是：

首先，必须在您的源程序中包含如下语句：

#include “C:\Art\PCH2542\INCLUDE\PCH2542.H”

注：以上语句采用默认路径和默认板号，应根据您的板号和安装情况确定 PCH2542.H 文件的正确路径，当然也可以把此文件拷到您的源程序目录中。

另外，要在 VB 环境中用子线程以实现高速、连续数据采集与存盘，请务必使用 VB5.0 版本。当然如果您有 VB6.0 的最新版，也可以实现子线程操作。

Visual Basic:

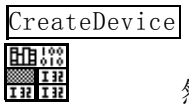
要使用如下函数一个关键的问题是首先必须将我们提供的模块文件(*.Bas)加入到您的 VB 工程中。其方法是选择

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 PCH2542.Bas 模块文件，该文件的路径为用户安装驱动程序后其子目录 Samples\VB 下面。

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，我们不能保证完全顺利运行。

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 的进一步介绍请见本文最后一部分关于 LabView 的专述。其驱动程序接口单元模块的使用方法如下：

一、在 LabView 中打开 PCH2542.VI 文件，用鼠标单击接口单元图标，比如 CreateDevice 图标 

然后按 Ctrl+C 或选择 LabView 菜单 Edit 中的 Copy 命令，接着进入用户的应用程序 LabView 中，按 Ctrl+V 或选择 LabView 菜单 Edit 中的 Paste 命令，即可将接口单元加入到用户工程中，然后按以下函数原型说明或演示程序的说明连接该接口模块即可顺利使用。

二、根据LabView语言本身的规定，接口单元图标以黑色的较粗的中间线为中心，以左边的方格为数据输入端，右边的方格为数据的输出端，如ReadDeviceProAD接口单元，设备对象句柄、用户分配的数据缓冲区、要求采集的数据长度等信息从接口单元左边输入端进入单元，待单元接口被执行后，需要返回给用户的数据从接口单元右边的输出端输出，其他接口完全同理。

三、在单元接口图标中，凡标有“I32”为有符号长整型 32 位数据类型，“U16”为无符号短整型 16 位数据类型，“[U16]”为无符号 16 位短整型数组或缓冲区或指针，“[U32]”与“[U16]”同理，只是位数不一样。

第二节、设备对象管理函数原型说明

♦ **创建设备对象函数（逻辑号）**

函数原型：

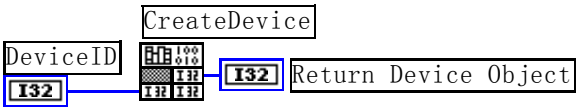
Visual C++：

HANDLE CreateDevice(int DeviceLgcID = 0)

Visual Basic：

Declare Function CreateDevice Lib "PCH2542" (Optional ByVal DeviceLgcID As Long = 0) As Long

LabVIEW：



功能：该函数使用逻辑号创建设备对象，并返回其设备对象句柄 hDevice。只有成功获取 hDevice，您才能实现对该设备所有功能的访问。

参数：

DeviceID 设备 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 DeviceID 标识值为名称后缀的标识符来确认和管理该设备。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。

相关函数： CreateDevice　GetDeviceCount　GetDeviceCurrentID　ListDeviceDlg　ReleaseDevice

Visual C++程序举例

：

```
HANDLE hDevice;     // 定义设备对象句柄
int DeviceLgcID = 0;
```

```
hDevice = PCH2542_CreateDevice (DeviceLgcID); // 创建设备对象,并取得设备对象句柄
if(hDevice == INVALIDE_HANDLE_VALUE); // 判断设备对象句柄是否有效
  {
    return;    // 退出该函数
 }
    :
```

Visual Basic 程序举例

```
    :
Dim hDevice As Long   ' 定义设备对象句柄
Dim DeviceLgcID As Long
DeviceLgcID = 0
hDevice = PCH2542_CreateDevice (DeviceLgcID)   ' 创建设备对象,并取得设备对象句柄
If hDevice = INVALID_HANDLE_VALUE Then   ' 判断设备对象句柄是否有效
    MsgBox “创建设备对象失败”
    Exit Sub      ' 退出该过程
End If
    :
```

♦ **取得本计算机系统中 PCH2542 设备的总数量**

函数原型：

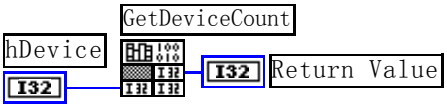
Visual C++:

int GetDeviceCount (HANDLE hDevice)

Visual Basic:

Declare Function GetDeviceCount Lib "PCH2542" (ByVal hDevice As Long) As Long

LabVIEW：



功能：取得 PCH2542 设备的数量。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

返回值：返回系统中 PCH2542 的数量。

相关函数： CreateDevice　GetDeviceCount　GetDeviceCurrentID　ListDeviceDlg　ReleaseDevice

♦ **取得该设备当前逻辑 ID 和物理 ID**

函数原型：

Visual C++:

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

Visual Basic:

```
Declare Function GetDeviceCurrentID Lib "PCH2542" (_
                                    ByVal hDevice As Long，_
                                    ByRef DeviceLgcID As Long,_
                                    ByRef DevicePhysID As Long) As Boolean
```

LabVIEW：

请参考相关演示程序。

功能：取得指定设备逻辑和物理 ID 号。

参数：

hDevice 设备对象句柄，它指向要取得逻辑和物理号的设备，它应由CreateDevice创建。
DeviceLgcID 返回设备的逻辑 ID，它的取值范围为[0, 15]。
DevicePhysID 返回设备的物理 ID，它的取值范围为[0, 15]，它的具体值由卡上的拨码器 DID 决定。
返回值：如果初始化设备对象成功，则返回TRUE，否则返回FALSE，用户可用GetLastErrorEx捕获当前错误码，并加以分析。
相关函数： CreateDevice GetDeviceCount GetDeviceCurrentID ListDeviceDlg ReleaseDevice

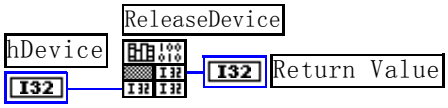
♦ **用对话框控件列表计算机系统中所有 PCH2542 设备各种配置信息**

函数原型：
Visual C++:
BOOL ListDeviceDlg (HANDLE hDevice)
Visual Basic:
Declare Function ListDeviceDlg Lib "PCH2542" (ByVal hDevice As Long) As Boolean
LabVIEW：
请参考相关演示程序。

功能：列表系统中 PCH2542 的硬件配置信息。
参数：
hDevice设备对象句柄，它应由CreateDevice创建。
返回值：若成功，则弹出对话框控件列表所有 PCH2542 设备的配置情况。
相关函数： CreateDevice 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++:
BOOL ReleaseDevice (HANDLE hDevice)
Visual Basic:
Declare Function ReleaseDevice Lib "PCH2542" (ByVal hDevice As Long) As Boolean
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，用户可以用GetLastErrorEx捕获错误码。
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后，再一次执行这些函数前，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DMA控制器、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。

第二节、计数器控制函数原型说明

♦ **启动设备计数器**

函数原型：
Viusal C++:
BOOL StartDeviceCNT (HANDLE hDevice,
ULONG nChannel)
Visual Basic:
Declare Function PCH2542_StartDeviceCNT Lib "PCH2542" (ByVal hDevice As Long, _

ByVal nChannel As Long) As Boolean

LabVIEW:
请参考相关演示程序。

功能：启动设备计数器。
参数：
hDevice设备对象句柄，它应由CreateDevice创建。
nChannel 计数器通道选择[0--8]。
返回值：若成功，返回 TRUE，否则返回 FALSE。
相关函数： CreateDevice StartDeviceCNT SetDeviceCNT
GetDeviceCNT StopDeviceCNT ReleaseDevice

- **设置计数器的初值**

函数原型：
Viusal C++:

```
BOOL SetDeviceCNT (HANDLE hDevice,
                   ULONG CNTVal,
                   ULONG ControlMode,
                   ULONG nChannel)
```

Visual Basic:

```
Declare Function SetDeviceCNT Lib "PCH2542" ( _
                                  ByVal hDevice As Long, _
                                  ByVal CNTVal As Long, _
                                  ByVal ControlMode As Long, _
                                  ByVal nChannel As Long) As Boolean
```

LabVIEW:
请参考相关演示程序。

功能：设置计数器的初值。
参数：
hDevice设备对象句柄，它应由CreateDevice创建。
CNTVal 计数初值
ControlMode 方式控制字
nChannel 计数器通道选择[0--8]。
返回值：若成功，返回TRUE，否则返回FALSE。
相关函数： CreateDevice StartDeviceCNT SetDeviceCNT
GetDeviceCNT StopDeviceCNT ReleaseDevice

- **取得各路计数器的当前计数值**

函数原型：
Viusal C++ :

```
BOOL GetDeviceCNT (HANDLE hDevice,
                   PULONG pCNTVal,
                   PULONG pControlMode,
                   ULONG nChannel)
```

Visual Basic:

```
Declare FunctionPCH2542_GetDeviceCNT Lib "PCH2542" ( _
                                  ByVal hDevice As Long, _
                                  ByRef pCNTVal As Long, _
                                  ByRef pControlMode As Long, _
```

ByVal nChannel As Long) As Boolean

LabVIEW:

请参考相关演示程序。

功能：取得各路计数器的当前计数值。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

pCNTVal 计数初值。

pControlMode 方式控制字

ControlMode 控制字模式

常量名	常量值	功能定义
PCH2542_GATEMODE_POSITIVE_0	0x00	计数方式 0，计数器结束中断方式
PCH2542_GATEMODE_RISING_1	0x01	计数方式 1，可编程单次脉冲方式
PCH2542_GATEMODE_POSITIVE_2	0x02	计数方式 2，频率发生器方式
PCH2542_GATEMODE_POSITIVE_3	0x03	计数方式 3，方波频率发生器方式
PCH2542_GATEMODE_POSITIVE_4	0x04	计数方式 4，软件触发选通方式
PCH2542_GATEMODE_RISING_5	0x05	计数方式 5，硬件触发选通方式

nChannel 计数器通道选择[0--8]。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数：CreateDevice StartDeviceCNT SetDeviceCNT GetDeviceCNT StopDeviceCNT ReleaseDevice

♦ **停止设备计数**

函数原型：

Viusal C++ :

BOOL StopDeviceCNT (HANDLE hDevice,
ULONG nChannel)

Visual Basic:

Declare Function PCH2542_StopDeviceCNT Lib "PCH2542" (_
ByVal hDevice As Long, _
ByVal nChannel As Long) As Boolean

LabVIEW:

请参考相关演示程序。

功能：停止设备计数。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

nChannel 计数器通道选择[0--8]。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数：CreateDevice StartDeviceCNT SetDeviceCNT GetDeviceCNT StopDeviceCNT ReleaseDevice

第四章 上层用户函数接口应用实例

第一节、简易程序演示说明

一、怎样使用SetDeviceCNT函数设置计数器初值

Visual C++:

其详细应用实例及正确代码请参考 Visual C++测试与演示系统，您先点击 Windows 系统的[开始]菜单，再按下列顺序点击，即可打开基于 VC 的 Sys 工程。

[程序]丨[阿尔泰测控演示系统]丨[PCH2542 9 路 CNT 卡]丨[Microsoft Visual C++]丨[简易代码演示]

第二节、高级程序演示说明

高级程序演示了本设备的所有功能，您先点击 Windows 系统的[开始]菜单，再按下列顺序点击，即可打开基于 VC 的 Sys 工程(主要参考 PCH2542.h 和 ADDoc.cpp)。

[程序]丨**[阿尔泰测控演示系统]**丨**[PCH2542 9 路 CNT 卡]**丨**[Microsoft Visual C++]**丨[高级代码演示]

其默认存放路径为：系统盘\ART\PCH2542\SAMPLES\VC\ADVANCED

其他语言的演示可以用上面类似的方法找到。

第五章 共用函数介绍

这部分函数不参与本设备的实际操作，它只是为您编写数据采集与处理程序时的有力手段，使您编写应用程序更容易，使您的应用程序更高效。

第一节、公用接口函数总列表（每个函数省略了前缀“PCH2542_”）

函数名	函数功能	备注
① PCH 总线内存映射寄存器操作函数		
GetDeviceAddr	取得指定 PCH 设备寄存器操作基地址	底层用户
GetDeviceBar	取得指定的指定设备寄存器组 BAR 地址	底层用户
GetDevVersion	获取设备固件及程序版本	底层用户
WriteRegisterByte	以字节(8Bit)方式写寄存器端口	底层用户
WriteRegisterWord	以字(16Bit)方式写寄存器端口	底层用户
WriteRegisterULong	以双字(32Bit)方式写寄存器端口	底层用户
ReadRegisterByte	以字节(8Bit)方式读寄存器端口	底层用户
ReadRegisterWord	以字(16Bit)方式读寄存器端口	底层用户
ReadRegisterULong	以双字(32Bit)方式读寄存器端口	底层用户
② ISA 总线 I/O 端口操作函数		
WritePortByte	以字节(8Bit)方式写 I/O 端口	用户程序操作端口
WritePortWord	以字(16Bit)方式写 I/O 端口	用户程序操作端口
WritePortULong	以无符号双字(32Bit)方式写 I/O 端口	用户程序操作端口
ReadPortByte	以字节(8Bit)方式读 I/O 端口	用户程序操作端口
ReadPortWord	以字(16Bit)方式读 I/O 端口	用户程序操作端口
ReadPortULong	以无符号双字(32Bit)方式读 I/O 端口	用户程序操作端口
③ 创建 Visual Basic 子线程，线程数量可达 32 个以上		
CreateSystemEvent	创建系统内核事件对象	用于线程同步或中断
ReleaseSystemEvent	释放系统内核事件对象	
④ 文件对象操作函数		
CreateFileObject	初始设备文件对象	
WriteFile	请求文件对象写用户数据到磁盘文件	
ReadFile	请求文件对象读数据到用户空间	
SetFileOffset	设置文件指针偏移	
GetFileLength	取得文件长度	
ReleaseFile	释放已有的文件对象	
GetDiskFreeBytes	取得指定磁盘的可用空间(字节)	适用于所有设备
⑤ 各种参数保存和读取函数		
SaveParaInt	保存整型参数到注册表	
LoadParaInt	从注册表中读取整型参数值	
SaveParaString	保存字符参数到注册表	
LoadParaString	从注册表中读取字符参数值	
⑥ 其他函数		

GetLastErrorEx	取得驱动函数错误信息	
RemoveLastErrorEx	移除指定函数的最后一次错误信息	

第二节、PCH 内存映射寄存器操作函数原型说明

♦ 取得指定内存映射寄存器的线性地址和物理地址

函数原型：

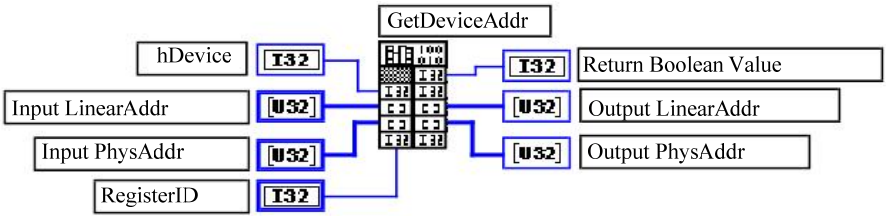
Visual C++:

```
BOOL GetDeviceAddr( HANDLE hDevice,
                    PULONG LinearAddr,
                    PULONG PhysAddr,
                    int RegisterID = 0)
```

Visual Basic:

```
Declare Function GetDeviceAddr Lib "PCH2542" (ByVal hDevice As Long, _
                                              ByRef LinearAddr As Long, _
                                              ByRef PhysAddr As Long, _
                                              Optional ByVal RegisterID As Integer = 0) As Boolean
```

LabVIEW:



功能：取得 PCH 设备指定的内存映射寄存器的线性地址。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

LinearAddr 指针参数，用于取得的映射寄存器指向的线性地址，RegisterID 指定的寄存器组属于 MEM 模式时该值不应为零，也就是说它可用于 WriteRegisterX 或 ReadRegisterX（X 代表 Byte、ULong、Word）等函数，以便于访问设备寄存器。它指明该设备位于系统空间的虚拟位置。但如果 RegisterID 指定的寄存器组属于 I/O 模式时该值通常为零，您不能通过以上函数访问设备。

PhysAddr 指针参数，用于取得的映射寄存器指向的物理地址，它指明该设备位于系统空间的物理位置。如果由 RegisterID 指定的寄存器组属于 I/O 模式，则可用于 WritePortX 或 ReadPortX（X 代表 Byte、ULong、Word）等函数，以便于访问设备寄存器。

RegisterID 指定映射寄存器的 ID 号，其取值范围为[0，5]，通常情况下，用户应使用 0 号映射寄存器，特殊情况下，我们为用户加以声明。本设备的寄存器组 ID 定义如下：

常量名	常量值	功能定义
PCH2542_REG_MEM_PLXCHIP	0x0000	0 号寄存器对应 PLX 芯片所使用的内存模式基地址（使用 LinearAddr）
PCH2542_REG_IO_PLXCHIP	0x0001	1 号寄存器对应 PLX 芯片所使用的 IO 模式基地址（PhysAddr）

返回值：如果执行成功，则返回 TRUE，它表明由 RegisterID 指定的映射寄存器的无符号 32 位线性地址和物理地址被正确返回，否则会返回 FALSE，同时还要检查其 LinearAddr 和 PhysAddr 是否为 0，若为 0 则依然视为失败。用户可用GetLastErrorEx捕获当前错误码，并加以分析。

相关函数: CreateDevice GetDeviceAddr WriteRegisterByte WriteRegisterWord WriteRegisterULong ReadRegisterByte ReadRegisterWord ReadRegisterULong ReleaseDevice

Visual C++程序举例:

```
        :
HANDLE hDevice;
ULONG LinearAddr, PhysAddr;
hDevice = CreateDevice(0);
if(!GetDeviceAddr(hDevice, &LinearAddr, &PhysAddr, 0))
{
    AfxMessageBox("取得设备地址失败…");
}
        :
```

Visual Basic 程序举例**:**

```
        :
Dim hDevice As Long
Dim LinearAddr, PhysAddr As Long
hDevice = CreateDevice(0)
if Not GetDeviceAddr(hDevice, LinearAddr,    PhysAddr, 0) then
    MsgBox "取得设备地址失败…"
End If
        :
```

- **取得指定的指定设备寄存器组 BAR 地址**

函数原型：

Visual C++:

```
BOOL GetDeviceBar (HANDLE hDevice,
                   ULONG pulPCHBar[6])
```

Visual Basic:

```
Declare Function GetDeviceBar Lib "PCH2542" (ByVal hDevice As Long, _
                                             ByVal pulPCHBar (0 to 5) As Long) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：取得指定的指定设备寄存器组 BAR 地址。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

pulPCHBar 返回 PCH BAR 所有地址。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数： CreateDevice ReleaseDevice

- **获取设备固件及程序版本**

函数原型：

Visual C++:

```
BOOL GetDevVersion (HANDLE hDevice,
                    PULONG pulFmwVersion,
                    PULONG pulDriverVersion)
```

Visual Basic:

```
Declare Function GetDevVersion Lib "PCH2542" (ByVal hDevice As Long, _
                                              ByRef pulFmwVersion As Long, _
                                              ByRef pulDriverVersion As Long) As Boolean
```

LabVIEW:

请参见相关演示程序。

功能：获取设备固件及程序版本。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

pulFmwVersion 指针参数，用于取得固件版本。

pulDriverVersion 指针参数，用于取得驱动版本。

返回值：如果执行成功，则返回 TRUE，否则会返回 FALSE。

相关函数: CreateDevice GetDeviceAddr WriteRegisterByte WriteRegisterWord WriteRegisterULong ReadRegisterByte ReadRegisterWord ReadRegisterULong ReleaseDevice

♦ **以单字节（即 8 位）方式写 PCH 内存映射寄存器的某个单元**

函数原型:

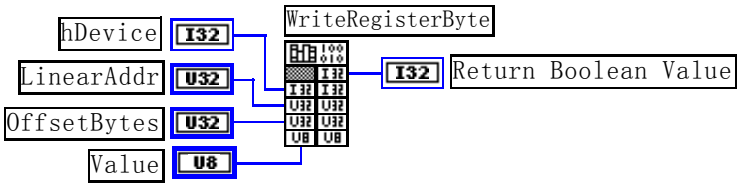
Visual C++:

```
BOOL WriteRegisterByte ( HANDLE hDevice,
                         ULONG LinearAddr,
                         ULONG OffsetBytes,
                         BYTE Value)
```

Visual Basic:

```
Declare Function WriteRegisterByte Lib "PCH2542" (ByVal hDevice As Long, _
                                                  ByVal LinearAddr As String, _
                                                  ByVal OffsetBytes As Long, _
                                                  ByVal Value As Byte ) As Boolean
```

LabVIEW:



功能：以单字节（即 8 位）方式写 PCH 内存映射寄存器。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

LinearAddr PCH设备内存映射寄存器的线性基地址，它的值应由GetDeviceAddr确定。

OffsetBytes相对于LinearAddr线性基地址的偏移字节数，它与LinearAddr两个参数共同确定WriteRegisterByte函数所访问的映射寄存器的内存单元。

Value 输出 8 位整数。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数: CreateDevice GetDeviceAddr WriteRegisterByte WriteRegisterWord WriteRegisterULong ReadRegisterByte ReadRegisterWord ReadRegisterULong ReleaseDevice

Visual C++程序举例:

```
    :
HANDLE hDevice;
ULONG LinearAddr, PhysAddr, OffsetBytes;
hDevice = CreateDevice(0)
if (!GetDeviceAddr(hDevice, &LinearAddr,   &PhysAddr, 0) )
{
    AfxMessageBox “取得设备地址失败…”;
}
OffsetBytes = 100;  // 指定操作相对于线性基地址偏移 100 个字节数位置的单元
WriteRegisterByte(hDevice, LinearAddr, OffsetBytes, 0x20); // 往指定映射寄存器单元写入 8 位的十六进制数据 20
ReleaseDevice( hDevice );   // 释放设备对象
```

：

Visual Basic 程序举例:

：

```
Dim hDevice As Long
Dim LinearAddr, PhysAddr, OffsetBytes As Long
hDevice = CreateDevice(0)
GetDeviceAddr( hDevice, LinearAddr, PhysAddr, 0)
OffsetBytes = 100
WriteRegisterByte( hDevice, LinearAddr, OffsetBytes, &H20)
ReleaseDevice(hDevice)
```

：

◆ **以双字节（即 16 位）方式写 PCH 内存映射寄存器的某个单元**

函数原型:

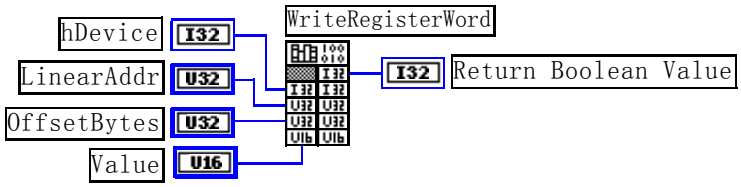
Visual C++:

```
BOOL WriteRegisterWord (HANDLE hDevice,
                        ULONG LinearAddr,
                        ULONG OffsetBytes,
                        WORD Value)
```

Visual Basic:

```
Declare Function WriteRegisterWord Lib "PCH2542" (ByVal hDevice As Long, _
                                                  ByVal LinearAddr As String, _
                                                  ByVal OffsetBytes As Long, _
                                                  ByVal Value As Integer) As Boolean
```

LabVIEW:



功能：以双字节（即 16 位）方式写 PCH 内存映射寄存器。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

LinearAddr PCH设备内存映射寄存器的线性基地址，它的值应由GetDeviceAddr确定。

OffsetBytes 相对于LinearAddr线性基地址的偏移字节数，它与LinearAddr两个参数共同确定WriteRegisterWord函数所访问的映射寄存器的内存单元。

Value 输出 16 位整型值。

返回值：无。

相关函数: CreateDevice GetDeviceAddr WriteRegisterByte WriteRegisterWord WriteRegisterULong ReadRegisterByte ReadRegisterWord ReadRegisterULong ReleaseDevice

Visual C++程序举例:

：

```
HANDLE hDevice;
ULONG LinearAddr, PhysAddr, OffsetBytes;
hDevice = CreateDevice(0)
if (!GetDeviceAddr(hDevice, &LinearAddr,   &PhysAddr, 0) )
{
    AfxMessageBox “取得设备地址失败…”;
}
```

```
OffsetBytes = 100;  // 指定操作相对于线性基地址偏移 100 个字节数位置的单元
WriteRegisterWord(hDevice, LinearAddr, OffsetBytes, 0x2000); //往指定映射寄存器单元写入 16 位的十六进制数据
ReleaseDevice( hDevice );   // 释放设备对象
    :
```

Visual Basic 程序举例:

```
    :
Dim hDevice As Long
Dim LinearAddr, PhysAddr, OffsetBytes As Long
hDevice = CreateDevice(0)
GetDeviceAddr( hDevice, LinearAddr, PhysAddr, 0)
OffsetBytes=100
WriteRegisterWord( hDevice, LinearAddr, OffsetBytes, &H2000)
ReleaseDevice(hDevice)
    :
```

♦ **以四字节（即 32 位）方式写 PCH 内存映射寄存器的某个单元**

函数原型:

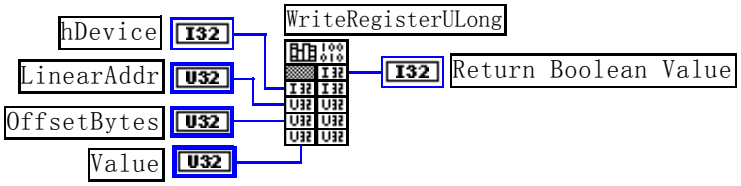
Visual C++:

```
BOOL WriteRegisterULong ( HANDLE hDevice,
                          ULONG LinearAddr,
                          ULONG OffsetBytes,
                          ULONG Value)
```

Visual Basic:

```
Declare Function WriteRegisterULong Lib "PCH2542" (ByVal hDevice As Long, _
                                                   ByVal LinearAddr As String, _
                                                   ByVal OffsetBytes As Long, _
                                                   ByVal Value As Long) As Boolean
```

LabVIEW:



功能：以四字节（即 32 位）方式写 PCH 内存映射寄存器。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

LinearAddr PCH设备内存映射寄存器的线性基地址，它的值应由GetDeviceAddr确定。

OffsetBytes 相对于LinearAddr线性基地址的偏移字节数，它与LinearAddr两个参数共同确定WriteRegisterULong函数所访问的映射寄存器的内存单元。

Value 输出 32 位整型值。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数: CreateDevice　GetDeviceAddr　WriteRegisterByte　WriteRegisterWord　WriteRegisterULong　ReadRegisterByte　ReadRegisterWord　ReadRegisterULong　ReleaseDevice

Visual C++程序举例:

```
    :
HANDLE hDevice;
ULONG LinearAddr, PhysAddr, OffsetBytes;
hDevice = CreateDevice(0)
if (!GetDeviceAddr(hDevice, &LinearAddr,   &PhysAddr, 0) )
```

```
{
    AfxMessageBox “取得设备地址失败…”;
}
OffsetBytes=100;  // 指定操作相对于线性基地址偏移 100 个字节数位置的单元
WriteRegisterULong(hDevice, LinearAddr, OffsetBytes, 0x20000000);  // 往指定映射寄存器单元写入 32 位的十六进制数据
ReleaseDevice( hDevice );  // 释放设备对象
    :
```

Visual Basic 程序举例:

```
    :
Dim hDevice As Long
Dim LinearAddr, PhysAddr, OffsetBytes As Long
hDevice = CreateDevice(0)
GetDeviceAddr( hDevice, LinearAddr, PhysAddr, 0)
OffsetBytes = 100
WriteRegisterULong( hDevice, LinearAddr, OffsetBytes, &H20000000)
ReleaseDevice(hDevice)
    :
```

◆ **以单字节（即 8 位）方式读 PCH 内存映射寄存器的某个单元**

函数原型:

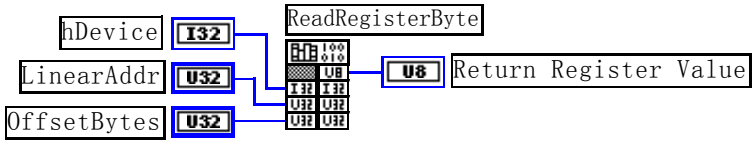
Visual C++:

```
BYTE ReadRegisterByte (HANDLE hDevice,
                       ULONG LinearAddr,
                       ULONG OffsetBytes)
```

Visual Basic:

```
Declare Function ReadRegisterByte Lib "PCH2542" (ByVal hDevice As Long, _
                                                 ByVal pbLinearAddr As String, _
                                                 ByVal OffsetBytes As Long) As Byte
```

LabVIEW:



功能：以单字节（即 8 位）方式读 PCH 内存映射寄存器的指定单元。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

LinearAddr PCH设备内存映射寄存器的线性基地址，它的值应由GetDeviceAddr确定。

OffsetBytes　相对于LinearAddr线性基地址的偏移字节数，它与LinearAddr两个参数共同确定 ReadRegisterByte函数所访问的映射寄存器的内存单元。

返回值：返回从指定内存映射寄存器单元所读取的 8 位数据。

相关函数:　CreateDevice　GetDeviceAddr　WriteRegisterByte　WriteRegisterWord　WriteRegisterULong　ReadRegisterByte　ReadRegisterWord　ReadRegisterULong　ReleaseDevice

Visual C++程序举例:

```
    :
HANDLE hDevice;
ULONG LinearAddr, PhysAddr, OffsetBytes;
BYTE Value;
hDevice = CreateDevice(0);  // 创建设备对象
```

```
GetDeviceAddr(hDevice, &LinearAddr, &PhysAddr, 0);    // 取得 PCH 设备 0 号映射寄存器的线性基地址
OffsetBytes = 100; // 指定操作相对于线性基地址偏移 100 个字节数位置的单元
Value = ReadRegisterByte(hDevice, LinearAddr, OffsetBytes);   // 从指定映射寄存器单元读入 8 位数据
ReleaseDevice( hDevice );   // 释放设备对象
    :
```

Visual Basic 程序举例:

```
    :
Dim hDevice As Long
Dim LinearAddr, PhysAddr, OffsetBytes As Long
Dim Value As Byte
hDevice = CreateDevice(0)
GetDeviceAddr( hDevice, Linearddr, PhysAddr, 0)
OffsetBytes = 100
Value = ReadRegisterByte( hDevice, LinearAddr, OffsetBytes)
ReleaseDevice(hDevice)
    :
```

♦ **以双字节（即 16 位）方式读 PCH 内存映射寄存器的某个单元**

函数原型:

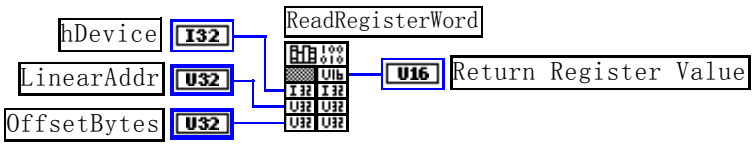
Visual C++:

```
WORD ReadRegisterWord( HANDLE hDevice,
                       ULONG LinearAddr,
                       ULONG OffsetBytes)
```

Visual Basic:

```
Declare Function ReadRegisterWord Lib "PCH2542" (ByVal hDevice As Long, _
                                                 ByVal LinearAddr As String, _
                                                 ByVal OffsetBytes As Long) As Integer
```

LabVIEW:



功能：以双字节（即 16 位）方式读 PCH 内存映射寄存器的指定单元。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

LinearAddr PCH设备内存映射寄存器的线性基地址，它的值应由GetDeviceAddr确定。

OffsetBytes　相对于LinearAddr线性基地址的偏移字节数，它与LinearAddr两个参数共同确定ReadRegisterWord函数所访问的映射寄存器的内存单元。

返回值：返回从指定内存映射寄存器单元所读取的 16 位数据。

相关函数:　CreateDevice　GetDeviceAddr　WriteRegisterByte
WriteRegisterWord　WriteRegisterULong　ReadRegisterByte
ReadRegisterWord　ReadRegisterULong　ReleaseDevice

Visual C++程序举例:

```
    :
HANDLE hDevice;
ULONG LinearAddr, PhysAddr, OffsetBytes;
WORD Value;
hDevice = CreateDevice(0);   // 创建设备对象
GetDeviceAddr(hDevice, &LinearAddr, &PhysAddr, 0);    // 取得 PCH 设备 0 号映射寄存器的线性基地址
OffsetBytes = 100; // 指定操作相对于线性基地址偏移 100 个字节数位置的单元
```

```
Value = ReadRegisterWord(hDevice, LinearAddr, OffsetBytes);   // 从指定映射寄存器单元读入 16 位数据
ReleaseDevice( hDevice );   // 释放设备对象
    :
```

Visual Basic 程序举例:

```
    :
Dim hDevice As Long
Dim LinearAddr, PhysAddr, OffsetBytes As Long
Dim Value As Word
hDevice = CreateDevice(0)
GetDeviceAddr( hDevice, Linearddr, PhysAddr, 0)
OffsetBytes = 100
Value = ReadRegisterWord( hDevice, LinearAddr, OffsetBytes)
ReleaseDevice(hDevice)
    :
```

♦ **以四字节（即 32 位）方式读 PCH 内存映射寄存器的某个单元**

函数原型:

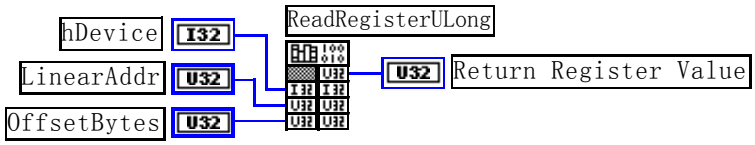
Visual C++:

```
ULONG ReadRegisterULong (HANDLE hDevice,
                         ULONG LinearAddr,
                         ULONG OffsetBytes)
```

Visual Basic:

```
Declare Function ReadRegisterULong Lib "PCH2542" (ByVal hDevice As Long, _
                                                  ByVal LinearAddr As String, _
                                                  ByVal OffsetBytes As Long) As Long
```

LabVIEW:



功能：以四字节（即 32 位）方式读 PCH 内存映射寄存器的指定单元。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

LinearAddr PCH设备内存映射寄存器的线性基地址，它的值应由GetDeviceAddr确定。

OffsetBytes 相对与LinearAddr线性基地址的偏移字节数，它与LinearAddr两个参数共同确定WriteRegisterULong函数所访问的映射寄存器的内存单元。

返回值：返回从指定内存映射寄存器单元所读取的 32 位数据。

相关函数: CreateDevice GetDeviceAddr WriteRegisterByte WriteRegisterWord WriteRegisterULong ReadRegisterByte ReadRegisterWord ReadRegisterULong ReleaseDevice

Visual C++程序举例:

```
    :
HANDLE hDevice;
ULONG LinearAddr, PhysAddr, OffsetBytes;
ULONG Value;
hDevice = CreateDevice(0);   // 创建设备对象
GetDeviceAddr(hDevice, &LinearAddr, &PhysAddr, 0);    // 取得 PCH 设备 0 号映射寄存器的线性基地址
OffsetBytes = 100;    // 指定操作相对于线性基地址偏移 100 个字节数位置的单元
Value = ReadRegisterULong(hDevice, LinearAddr, OffsetBytes);   // 从指定映射寄存器单元读入 32 位数据
ReleaseDevice( hDevice );   // 释放设备对象
```

：

Visual Basic 程序举例:

：

```
Dim hDevice As Long
Dim LinearAddr, PhysAddr, OffsetBytes As Long
Dim Value As Long
hDevice = CreateDevice(0)
GetDeviceAddr( hDevice, Linearddr, PhysAddr, 0)
OffsetBytes = 100
Value = ReadRegisterULong( hDevice, LinearAddr, OffsetBytes)
ReleaseDevice(hDevice)
```

：

第三节、I0 端口读写函数原型说明

注意：若您想在 WIN2K 系统的 User 模式中直接访问 I/O 端口，那么您可以安装光盘中 ISA\CommUser 目录下的公用驱动，然后调用其中的 WritePortByteEx 或 ReadPortByteEx 等有“Ex”后缀的函数即可。

♦ **以单字节(8Bit)方式写 I/O 端口**

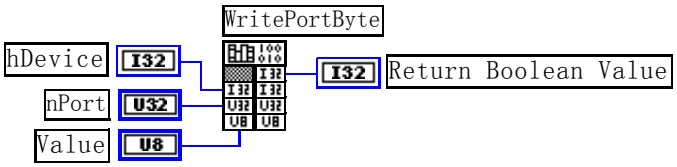
Visual C++:

```
BOOL WritePortByte (HANDLE hDevice,
                    UINT nPort,
                    BYTE Value)
```

Visual Basic:

```
Declare Function WritePortByte Lib "PCH2542" (ByVal hDevice As Long, _
                                              ByVal nPort As Long, _
                                              ByVal Value As Byte) As Boolean
```

LabVIEW:



功能：以单字节(8Bit)方式写 I/O 端口。

参数：

hDevice 设备对象句柄，它应由CreateDevice创建。

nPort 指定寄存器的物理基地址。

Value 写出的 8 位整型数据。

返回值：若成功，返回TRUE，否则返回FALSE，用户可用GetLastErrorEx捕获当前错误码。

相关函数： CreateDevice WritePortByte WritePortWord WritePortULong ReadPortByte ReadPortWord

♦ **以双字(16Bit)方式写 I/O 端口**

Visual C++:

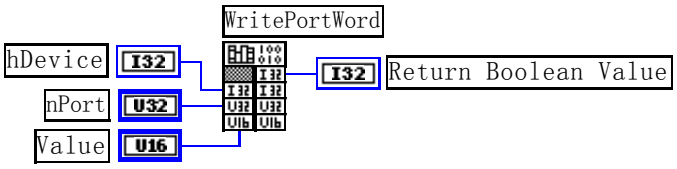
```
BOOL WritePortWord (HANDLE hDevice,
                    UINT nPort,
                    WORD Value)
```

Visual Basic:

```
Declare Function WritePortWord Lib "PCH2542" (ByVal hDevice As Long, _
                                              ByVal nPort As String, _
```

ByVal Value As Integer) As Boolean

LabVIEW:



功能：以双字(16Bit)方式写 I/O 端口。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

nPort 指定寄存器的物理基地址。

Value　写出的 8 位整型数据。

返回值：若成功，返回TRUE，否则返回FALSE，用户可用GetLastErrorEx捕获当前错误码。

相关函数： CreateDevice　WritePortByte　WritePortWord　WritePortULong　ReadPortByte　ReadPortWord

- **以四字节(32Bit)方式写 I/O 端口**

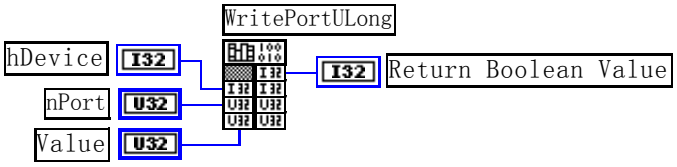
Visual C++:

```
BOOL WritePortULong (HANDLE hDevice,
                     UINT nPort,
                     ULONG Value)
```

Visual Basic:

```
Declare Function WritePortULong Lib "PCH2542" (ByVal hDevice As Long, _
                                               ByVal nPort As String, _
                                               ByVal Value As Long) As Boolean
```

LabVIEW:



功能：以四字节(32Bit)方式写 I/O 端口。

参数：

hDevice　设备对象句柄，它应由CreateDevice创建。

nPort 指定寄存器的物理基地址。

Value　写出的 8 位整型数据。

返回值：若成功，返回TRUE，否则返回FALSE，用户可用GetLastErrorEx捕获当前错误码。

相关函数： CreateDevice　WritePortByte　WritePortWord　WritePortULong　ReadPortByte　ReadPortWord

- **以单字节(8Bit)方式读 I/O 端口**

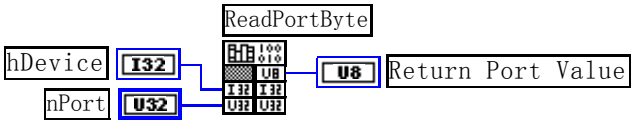
Visual C++:

```
BYTE ReadPortByte (HANDLE hDevice,
                   UINT nPort)
```

Visual Basic:

```
Declare Function ReadPortByte Lib "PCH2542" (ByVal hDevice As Long, _
                                             ByVal nPort As Long) As Byte
```

LabVIEW:



功能：以单字节(8Bit)方式读 I/O 端口。
参数：
hDevice设备对象句柄，它应由CreateDevice创建。
nPort 指定寄存器的物理基地址。
返回值：返回由 nPort 指定的端口的值。
相关函数： CreateDevice WritePortByte WritePortWord WritePortULong ReadPortByte ReadPortWord

- **以双字节(16Bit)方式读 I/O 端口**

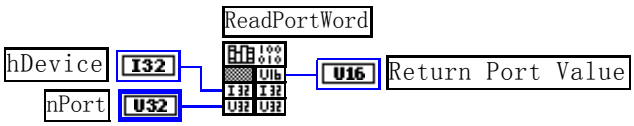
Visual C++:

```
WORD ReadPortWord (HANDLE hDevice,
                   UINT nPort)
```

Visual Basic:

```
Declare Function ReadPortWord Lib "PCH2542" (ByVal hDevice As Long, _
                                             ByVal nPort As LongWord) As Integer
```

LabVIEW:



功能：以双字节(16Bit)方式读 I/O 端口。
参数：
hDevice设备对象句柄，它应由CreateDevice创建。
nPort 指定寄存器的物理基地址。
返回值：返回由 nPort 指定的端口的值。
相关函数： CreateDevice WritePortByte WritePortWord WritePortULong ReadPortByte ReadPortWord

- **以四字节(32Bit)方式读 I/O 端口**

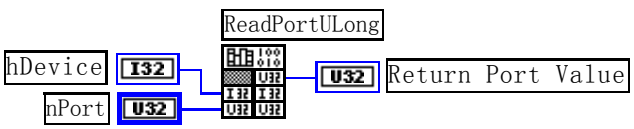
Visual C++:

```
ULONG ReadPortULong (HANDLE hDevice,
                     UINT nPort)
```

Visual Basic:

```
Declare Function ReadPortULong Lib "PCH2542" (ByVal hDevice As Long, _
                                              ByVal nPort As Long) As Long
```

LabVIEW:



功能：以四字节(32Bit)方式读 I/O 端口。
参数：
hDevice设备对象句柄，它应由CreateDevice创建。
nPort 指定寄存器的物理基地址。
返回值：返回由 nPort 指定端口的值。

相关函数： CreateDevice WritePortByte WritePortWord
WritePortULong ReadPortByte ReadPortWord

第四节、线程操作函数原型说明

（如果您的 VB6.0 中线程无法正常运行，可能是 VB6.0 语言本身的问题，请选用 VB5.0）

- **创建内核系统事件**

函数原型：

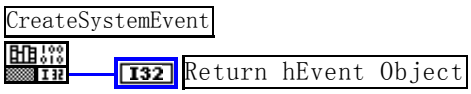
Visual C++:

HANDLE CreateSystemEvent(void)

Visual Basic:

Declare Function CreateSystemEvent Lib " PCH2542 " () As Long

LabVIEW:



功能：创建系统内核事件对象，它将被用于中断事件响应或数据采集线程同步事件。
参数：无任何参数。
返回值：若成功，返回系统内核事件对象句柄，否则返回－1(或 INVALID_HANDLE_VALUE)。

- **释放内核系统事件**

函数原型：

Visual C++:

BOOL ReleaseSystemEvent (HANDLE hEvent)

Visual Basic:

Declare Function ReleaseSystemEvent Lib " PCH2542 " (ByVal hEvent As Long) As Boolean

请参见相关演示程序。

功能：释放系统内核事件对象。
参数：
hEvent　被释放的内核事件对象。它应由CreateSystemEvent成功创建的对象。
返回值：若成功，则返回 TRUE。

第五节、文件对象操作函数原型说明

- **创建文件对象**

函数原型：

Visual C++:

```
HANDLE CreateFileObject (HANDLE hDevice,
                         LPCTSTR strFileName,
                         int Mode)
```

Visual Basic:

```
Declare Function CreateFileObject Lib "PCH2542" (ByVal hDevice As Long, _
                                                 ByVal strFileName As String, _
                                                 ByVal Mode As Integer) As Long
```

LabVIEW:

请参见相关演示程序。

功能：初始化设备文件对象，以期待 WriteFile 请求准备文件对象进行文件操作。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

strFileName 与新文件对象关联的磁盘文件名，可以包括盘符和路径等信息。在 C 语言中，其语法格式如："C:\\PCH2542\\Data.Dat"，在 Basic 中，其语法格式如："C:\PCH2542\Data.Dat"。

Mode 文件操作方式，所用的文件操作方式控制字定义如下(可通过或指令实现多种方式并操作)：

常量名	常量值	功能定义
PCH2542_modeRead	0x0000	只读文件方式
PCH2542_modeWrite	0x0001	只写文件方式
PCH2542_modeReadWrite	0x0002	既读又写文件方式
PCH2542_modeCreate	0x1000	如果文件不存在可以创建该文件，如果存在，则重建此文件，且清 0
PCH2542_typeText	0x4000	以文本方式操作文件

返回值：若成功，则返回文件对象句柄。

相关函数： CreateDevice CreateFileObject WriteFile ReadFile ReleaseFile ReleaseDevice

◆ **通过设备对象，往指定磁盘上写入用户空间的采样数据**

函数原型：

Visual C++:

```
BOOL WriteFile (HANDLE hFileObject,
                PVOID pDataBuffer,
                LONG nWriteSizeBytes)
```

Visual Basic:

```
Declare Function WriteFile Lib "PCH2542" (ByVal hFileObject As Long, _
                                          ByRef pDataBuffer As Byte, _
                                          ByVal nWriteSizeBytes As Long) As Boolean
```

LabVIEW:

详见相关演示程序。

功能：通过向设备对象发送"写磁盘消息"，设备对象便会以最快的速度完成写操作。注意为了保证写入的数据是可用的，这个操作将与用户程序保持同步，但与设备对象中的环形内存池操作保持异步，以得到更高的数据吞吐量，其文件名及路径应由CreateFileObject函数中的strFileName指定。

参数：

hFileObject 设备对象句柄，它应由CreateFileObject创建。

pDataBuffer 用户数据空间地址，可以是用户分配的数组空间。

nWriteSizeBytes 告诉设备对象往磁盘上一次写入数据的长度(以字节为单位)。

返回值：若成功，则返回TRUE，否则返回FALSE，用户可以用GetLastErrorEx捕获错误码。

相关函数： CreateFileObject WriteFile ReadFile ReleaseFile

◆ **通过设备对象,从指定磁盘文件中读采样数据**

函数原型：

Visual C++:

```
BOOL ReadFile (HANDLE hFileObject,
               PVOID pDataBuffer,
               LONG nOffsetBytes,
               LONG nReadSizeBytes)
```

Visual Basic:

```
Declare Function ReadFile Lib "PCH2542" ( ByVal hFileObject As Long, _
                                          ByRef pDataBuffer As Integer, _
                                          ByVal OffsetBytes As Long, _
                                          ByVal nReadSizeBytes As Long) As Boolean
```

LabVIEW:

详见相关演示程序。

功能：将磁盘数据从指定文件中读入用户内存空间中，其访问方式可由用户在创建文件对象时指定。
参数：
hFileObject 设备对象句柄，它应由CreateFileObject创建。
pDataBuffer 用于接受文件数据的用户缓冲区指针，可以是用户分配的数组空间。
OffsetBytes 指定从文件开始端所偏移的读位置。
nReadSizeBytes 告诉设备对象从磁盘上一次读入数据的长度(以字为单位)。
返回值：若成功，则返回TRUE，否则返回FALSE，用户可以用GetLastErrorEx捕获错误码。
相关函数： CreateFileObject WriteFile ReadFile ReleaseFile

- **设置文件偏移位置**

函数原型：
Visual C++:

```
BOOL SetFileOffset (HANDLE hFileObject,
                    LONG nOffsetBytes)
```

Visual Basic:

```
Declare Function SetFileOffset Lib "PCH2542" (ByVal hFileObject As Long,
                                              ByVal nOffsetBytes As Long) As Boolean
```

LabVIEW:
详见相关演示程序。

功能：设置文件偏移位置，用它可以定位读写起点。
参数：
hFileObject 文件对象句柄，它应由CreateFileObject创建。
返回值：若成功，则返回TRUE，否则返回FALSE，用户可以用GetLastErrorEx捕获错误码。
相关函数： CreateFileObject WriteFile ReadFile ReleaseFile

- **取得文件长度（字节）**

函数原型：
Visual C++:

```
ULONG GetFileLength (HANDLE hFileObject)
```

Visual Basic:

```
Declare Function GetFileLength Lib "PCH2542" (ByVal hFileObject As Long) As Long
```

LabVIEW:
详见相关演示程序。

功能：取得文件长度。
参数：
hFileObject 设备对象句柄，它应由CreateFileObject创建。
返回值：若成功，则返回>1，否则返回 0，用户可以用GetLastErrorEx捕获错误码。
相关函数： CreateFileObject WriteFile ReadFile ReleaseFile

- **释放设备文件对象**

函数原型：
Visual C++:

```
BOOL ReleaseFile (HANDLE hFileObject)
```

Visual Basic:

Declare Function ReleaseFile Lib "PCH2542" (ByVal hFileObject As Long) As Boolean

LabVIEW:

详见相关演示程序。

功能：释放设备文件对象。

参数：

hFileObject 设备对象句柄，它应由CreateFileObject创建。

返回值：若成功，则返回TRUE，否则返回FALSE，用户可以用GetLastErrorEx捕获错误码。

相关函数： CreateFileObject WriteFile ReadFile ReleaseFile

- **取得指定磁盘的可用空间**

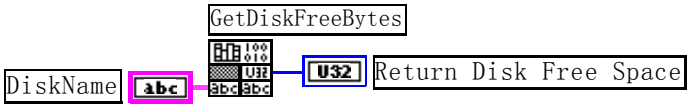
Visual C++:

ULONGLONG GetDiskFreeBytes (LPCTSTR strDiskName)

Visual Basic:

Declare Function GetDiskFreeBytes Lib "PCH2542" (ByVal DiskName As String) As Currency

LabVIEW:



功能：取得指定磁盘的可用剩余空间(以字为单位)。

参数：

DiskName 需要访问的盘符，若为 C 盘为"C:\\"，D 盘为"D:\\"，以此类推。

返回值：若成功，返回大于或等于 0 的长整型值，否则返回零值，用户可用GetLastErrorEx捕获错误码。注意使用 64 位整型变量。

第六节、各种参数保存和读取函数原型说明

- **将整型变量的参数值保存在系统注册表中**

函数原型：

Visual C++:

```
BOOL SaveParaInt(HANDLE hDevice,
                 LPCTSTR strParaName,
                 int nValue)
```

Visual Basic:

```
Declare Function SaveParaInt Lib "PCH2542" (ByVal hDevice As Long, _
                                            ByVal strParaName As String, _
                                            ByVal nValue As Integer) As Boolean
```

LabVIEW:

详见相关演示程序。

功能：将整型变量的参数值保存在系统注册表中。具体保存位置视设备逻辑号而定。如逻辑号为“0”的其他参数保存位置为：HKEY_CURRENT_USER\Software\Art\PCH2542\Device-0\Others。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

strParaName 整型参数字符名。它指名该参数在注册表中的字符键项。

nValue 整型参数值。它保存在由 strParaName 命名的键项里。

返回值：若成功，则返回TRUE，否则返回FALSE，用户可以用GetLastErrorEx捕获错误码。

相关函数： SaveParaInt LoadParaInt SaveParaString LoadParaString

- **将整型变量的参数值从系统注册表中读出**

函数原型：

Visual C++:

```
UINT LoadParaInt(HANDLE hDevice,
                 LPCTSTR strParaName,
                 int nDefaultVal)
```

Visual Basic:

```
Declare Function LoadParaInt Lib "PCH2542" (ByVal hDevice As Long, _
                                            ByVal strParaName As String, _
                                            ByVal nDefaultVal As Integer) As Long
```

LabVIEW:

详见相关演示程序。

功能：将整型变量的参数值从系统注册表中读出。读出参数值的具体位置视设备逻辑号而定。如逻辑号为“0”的其他参数保存位置为：HKEY_CURRENT_USER\Software\Art\PCH2542\Device-0\Others。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

strParaName 整型参数字符名。它指名该参数在注册表中的字符键项。

nDefaultVal 若 strParaName 指定的键项不存在，则由该参数指定的默认值返回。

返回值：若指定的整型参数项存在，则返回其整型值。否则返回由 nDefaultVal 指定的默认值。

相关函数： SaveParaInt LoadParaInt SaveParaString LoadParaString

- **将字符变量的参数值保存在系统注册表中**

函数原型：

Visual C++:

```
BOOL SaveParaString (HANDLE hDevice,
                     LPCTSTR strParaName,
                     LPCTSTR strParaVal)
```

Visual Basic:

```
Declare Function SaveParaString Lib "PCH2542" (ByVal hDevice As Long, _
                                               ByVal strParaName As String, _
                                               ByVal strParaVal As String) As Boolean
```

LabVIEW:

详见相关演示程序。

功能：将整型变量的参数值保存在系统注册表中。具体保存位置视设备逻辑号而定。如逻辑号为“0”的其他参数保存位置为：HKEY_CURRENT_USER\Software\Art\PCH2542\Device-0\Others。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

strParaName 整型参数字符名。它指名该参数在注册表中的字符键项。

strParaVal 字符参数值。它保存在由 strParaName 命名的键项里。

返回值：若成功，则返回TRUE，否则返回FALSE，用户可以用GetLastErrorEx捕获错误码。

相关函数： SaveParaInt LoadParaInt SaveParaString LoadParaString

- **将字符变量的参数值从系统注册表中读出**

函数原型：

Visual C++:

```
BOOL LoadParaString (HANDLE hDevice,
                     LPCTSTR strParaName,
                     LPCTSTR strParaVal,
                     LPCTSTR strDefaultVal)
```

Visual Basic:

```
Declare Function LoadParaString Lib "PCH2542" (ByVal hDevice As Long, _
                                               ByVal strParaName As String, _
                                               ByVal strParaVal As String, _
                                               ByVal strDefaultVal As String) As Boolean
```

LabVIEW:

详见相关演示程序。

功能：将字符变量的参数值从系统注册表中读出。读出参数值的具体位置视设备逻辑号而定。如逻辑号为“0”的其他参数保存位置为：HKEY_CURRENT_USER\Software\Art\PCH2542\Device-0\Others。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

strParaName 字符参数字符名。它指名该参数在注册表中的字符键项。

strParaVal 取得 strParaName 指定的键项的字符值。

strDefaultVal 若 strParaName 指定的键项不存在，则由该参数指定的默认值返回。

返回值：若成功，则返回TRUE，否则返回FALSE，用户可以用GetLastErrorEx捕获错误码。

相关函数： SaveParaInt LoadParaInt SaveParaString LoadParaString

第七节、其他函数原型说明

- **怎样获取驱动函数错误信息**

函数原型：

Visual C++:

```
DWORD GetLastErrorEx (LPCTSTR strFuncName,
                      LPTSTR strErrorMsg)
```

Visual Basic:

```
Declare Function GetLastErrorEx Lib "PCH2542" (ByVal strFuncName As String,_
                                               ByVal strErrorMsg As String）As Long
```

LabVIEW:

详见相关演示程序。

功能：将当某个驱动函数出错时，可以调用此函数获得具体的错误和错误信息字串。

参数：

strFuncName 出错函数的名称，注意大小写。注意此函数必须是完整名称。如 AD 初始化函数 PCH2542_InitDeviceAD 出现错误，此时调用该函数时，此参数必须为“PCH2542_InitDeviceAD”，否则得不到相应信息。

strErrorMsg 取得指定函数的错误信息串，该串为字符数组，其分配空间最好不要小于 256 字节。

返回值：返回错误码。

相关函数：无。

- **移除驱动函数错误信息**

函数原型：

Visual C++:

```
BOOL RemoveLastErrorEx ( LPCTSTR strFuncName）
```

Visual Basic:

```
Declare Function RemoveLastErrorEx Lib "PCH2542" (ByVal strFuncName  As String）As Boolean
```

LabVIEW:

详见相关演示程序。

功能：从错误信息库中移除指定函数的最后一次错误信息。

参数：

strFuncName 出错函数的名称，注意大小写。注意此函数必须是完整名称。如 AD 初始化函数

PCH2542_InitDeviceAD 出现错误，此时调用该函数时，此参数必须为“PCH2542_InitDeviceAD”，否则得不到相应信息。

返回值：若成功，则返回TRUE，否则返回FALSE，用户可以用GetLastErrorEx 捕获错误码。

相关函数：无。