PCIe2321 高驱动 DIO 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

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 PCIexxxx_则被省略。如 PCIe2321_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 使用纲要

第一节、如何实现数字量的简便操作

当您有了hDevice设备对象句柄后，便可用SetDeviceDO函数实现数字量的输出操作，其各路数字量的输出状态由其bDOSts[48]中的相应元素决定。由GetDeviceDI函数实现数字量的输入操作，其各路数字量的输入状态由其bDISts[48]中的相应元素决定。

第二节、哪些函数对您不是必须的

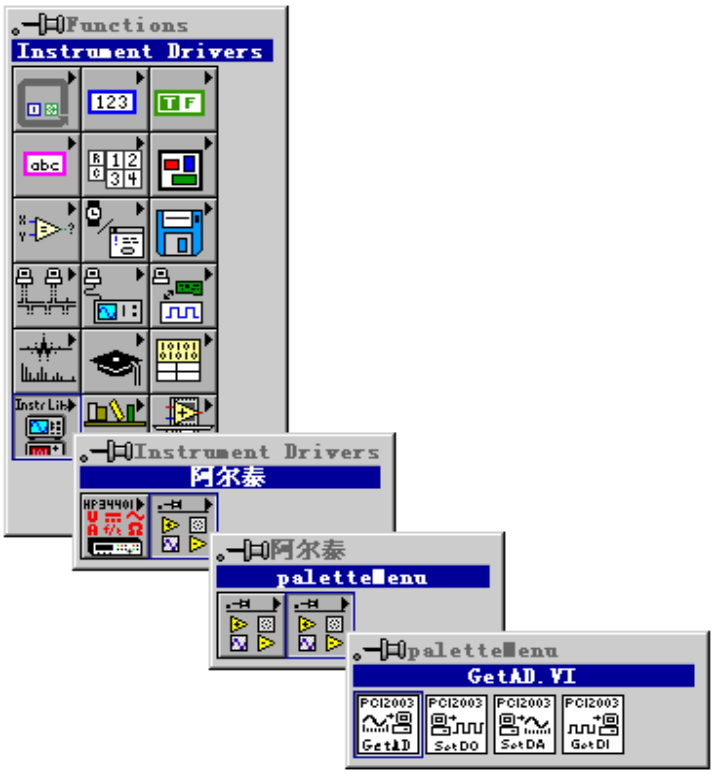
公共函数如CreateFileObject，WriteFile，ReadFile等一般来说都是辅助性函数，除非您要使用存盘功能。如果您使用上层用户函数访问设备，那么GetDeviceAddr，WriteRegisterByte，WriteRegisterWord，WriteRegisterULong，ReadRegisterByte，ReadRegisterWord，ReadRegisterULong等函数您可完全不必理会，除非您是作为底层用户管理设备。而WritePortByte，WritePortWord，WritePortULong，ReadPortByte，ReadPortWord，ReadPortULong则对PCIe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 PCIe 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PCIe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PCIe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“PCIe2321_”）

函数名	函数功能	备注
① 设备对象操作函数		
CreateDevice	创建 PCIe 设备对象(用设备逻辑号)	上层及底层用户
GetDeviceCount	取得同一种 PCIe 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 PCIe 设备的各种配置	上层及底层用户
ReleaseDevice	关闭设备，且释放 PCIe 总线设备对象	上层及底层用户
② 开关量控制函数 对 P1 操作		
EnableStsP1DIO	设置开关量输入或输出状态	上层用户
SetDeviceDO_P1A	输出开关量状态	上层用户
GetDeviceDI_P1A	输入开关量状态	上层用户
SetDeviceDO_P1B	输出开关量状态	上层用户
GetDeviceDI_P1B	输入开关量状态	上层用户
SetDeviceDO_P1C	输出开关量状态	上层用户
GetDeviceDI_P1C	输入开关量状态	上层用户
③ 开关量控制函数 对 P2 操作		
EnableStsP2DIO	设置开关量输入或输出状态	上层用户
SetDeviceDO_P2A	输出开关量状态	上层用户
GetDeviceDI_P2A	输入开关量状态	上层用户
SetDeviceDO_P2B	输出开关量状态	
GetDeviceDI_P2B	输入开关量状态	上层用户
ReleaseDeviceDmaAD	释放设备上的 AD 部件	上层用户
SetDeviceDO_P2C	输出开关量状态	
GetDeviceDI_P2C	输入开关量状态	
④ 计数器控制函数		
SetDevCNTInitValue	设置计数器初值	上层用户
EnableDevCNT	是否使能计数器	上层用户

GetDevCNTVal	取得所有计数器的计数值	上层用户
ResetDevCNT	计数器复位清零	上层用户
⑤ 中断函数		
InitDeviceInt	初始化中断	上层用户
GetDeviceIntCount	在中断初始化后，用它取得中断服务程序产生的次数	上层用户
ReleaseDeviceInt	释放中断资源	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\PCIe2321\INCLUDE\PCIe2321.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 PCIe2321.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 PCIe2321.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 PCIe2321.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：

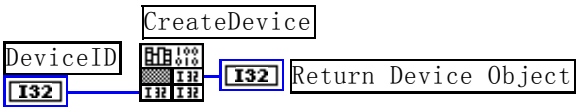
Visual C++：

HANDLE CreateDevice (int DeviceID = 0)

Visual Basic：

Declare Function PCIE2321_CreateDevice Lib "PCIE2321_32" (ByVal DeviceLgcID As Long) 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 = PCIe2321_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 = PCIe2321_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 PCIe2321 设备的总数量

函数原型：

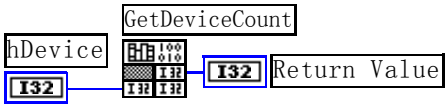
Visual C++:

int GetDeviceCount (HANDLE hDevice)

Visual Basic:

Declare Function PCIE2321_GetDeviceCount Lib "PCIE2321_32" (ByVal hDevice As Long) As Integer

LabVIEW：



功能：取得 PCIe2321 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 PCIe2321 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 PCIE2321_GetDeviceCurrentID Lib "PCIE2321_32" (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拔码器 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 PCIe2321 设备各种配置信息**

函数原型：

Visual C++:

```
BOOL ListDeviceDlg (HANDLE hDevice)
```

Visual Basic:

```
Declare Function PCIE2321_ListDeviceDlg Lib "PCIE2321_32" (ByVal hDevice As Long) As Boolean
```

LabVIEW：

请参考相关演示程序。

功能：列表系统中 PCIe2321 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 PCIe2321 设备的配置情况。

相关函数： CreateDevice GetDeviceCount GetDeviceCurrentID ListDeviceDlg ReleaseDevice

♦ **释放设备对象所占的系统资源及设备对象**

函数原型：

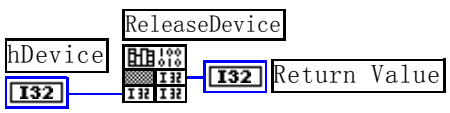
Visual C++:

```
BOOL ReleaseDevice (HANDLE hDevice)
```

Visual Basic:

```
Declare Function PCIE2321_ReleaseDevice Lib "PCIE2321_32" (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 GetDeviceCount GetDeviceCurrentID ListDeviceDlg ReleaseDevice

第三节、开关量控制函数

对 P1 操作

♦ **设置开关量输入或输出状态**

函数原型：

Visual C++：

```
BOOL EnableStsP1DIO (HANDLE hDevice,
                     BOOL bExtTrig,
                     BOOL bP1Sts[3])
```

Visual Basic:

```
Declare Function PCIE2321_EnableStsP1DIO Lib "PCIE2321_32" (_
                                    ByVal hDevice As Long,_
                                    ByVal bExtTrig As Boolean,_
                                    ByRef bP1Sts As Long) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：设置开关量输入或输出状态。
参数：
hDevice 设备对象句柄，它应由设备的CreateDevice创建。
bExtTrig 是否使能外部触发功能 TRUE：使能，FALSE：禁止。
bP1Sts[3] bP1Sts[0]bP1Sts[2]分别控制 P1A、P1B、P1C 端口 FALSE：开关量输入，TRUE：开关量输出
返回值：若成功返回 FALSE，否则返回 TRUE。
相关函数： CreateDevice EnableStsP1DIO SetDeviceDO_P1A GetDeviceDI_P1A SetDeviceDO_P1B GetDeviceDI_P1B SetDeviceDO_P1C GetDeviceDI_P1C ReleaseDevice

对 P1A 操作

♦**开关量输出状态**

函数原型：

Visual C++：

```
BOOL SetDeviceDO_P1A (HANDLE hDevice,
                      BYTE bDIOSts[8])
```

Visual Basic:

```
Declare Function PCIE2321_SetDeviceDO_P1A Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                    ByRef bDIOsts As Byte) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：开关量输出状态。
参数：
hDevice 设备对象句柄，它应由设备的CreateDevice创建。
bDIOSts[8] bDIOSts 参数(注意: 必须定义为 8 个字节元素的数组)。
返回值：若成功返回 FALSE，否则返回 TRUE。
相关函数： CreateDevice EnableStsP1DIO SetDeviceDO_P1A GetDeviceDI_P1A SetDeviceDO_P1B GetDeviceDI_P1B

SetDeviceDO_P1C　　GetDeviceDI_P1C　　ReleaseDevice

♦开关量输入状态

函数原型：

Visual C++：

```
BOOL GetDeviceDI_P1A (HANDLE hDevice,
                      BYTE bDIOSts[8])
```

Visual Basic:

```
Declare Function PCIE2321_GetDeviceDI_P1A Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                             ByRef bDIOsts As Byte) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：开关量输入状态。

参数：

hDevice 设备对象句柄，它应由设备的CreateDevice创建。

bDIOSts[8] bDIOSts 参数(注意: 必须定义为 8 个字节元素的数组)。

返回值：若成功返回 FALSE，否则返回 TRUE。

相关函数: CreateDevice　　EnableStsP1DIO　　SetDeviceDO_P1A
GetDeviceDI_P1A　　SetDeviceDO_P1B　　GetDeviceDI_P1B
SetDeviceDO_P1C　　GetDeviceDI_P1C　　ReleaseDevice

对 P1B 操作

♦开关量输出状态

函数原型：

Visual C++：

```
BOOL SetDeviceDO_P1B (HANDLE hDevice,
                      BYTE bDIOSts[8])
```

Visual Basic:

```
Declare Function PCIE2321_SetDeviceDO_P1B Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                             ByRef bDIOsts As Byte) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：开关量输出状态。

参数：

hDevice 设备对象句柄，它应由设备的CreateDevice创建。

bDIOSts[8] bDIOSts 参数(注意: 必须定义为 8 个字节元素的数组)。

返回值：若成功返回 FALSE，否则返回 TRUE。

相关函数: CreateDevice　　EnableStsP1DIO　　SetDeviceDO_P1A
GetDeviceDI_P1A　　SetDeviceDO_P1B　　GetDeviceDI_P1B
SetDeviceDO_P1C　　GetDeviceDI_P1C　　ReleaseDevice

♦开关量输入状态

函数原型：

Visual C++：

```
BOOL GetDeviceDI_P1B (HANDLE hDevice,
                      BYTE bDIOSts[8])
```

Visual Basic:

```
Declare Function PCIE2321_GetDeviceDI_P1B Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                             ByRef bIDOsts As Byte) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：开关量输入状态。
参数：
hDevice 设备对象句柄，它应由设备的CreateDevice创建。
bDIOSts[8] bDIOSts 参数(注意: 必须定义为 8 个字节元素的数组)。
返回值：若成功返回 FALSE，否则返回 TRUE。
相关函数: CreateDevice EnableStsP1DIO SetDeviceDO_P1A GetDeviceDI_P1A SetDeviceDO_P1B GetDeviceDI_P1B SetDeviceDO_P1C GetDeviceDI_P1C ReleaseDevice

对 P1C 操作

♦开关量输出状态

函数原型：

Visual C++：

```
BOOL SetDeviceDO_P1C(HANDLE hDevice,
                     BYTE bDIOSts[8])
```

Visual Basic:

```
Declare Function PCIE2321_SetDeviceDO_P1C Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                    ByRef bDIOsts As Byte) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：开关量输出状态。
参数：
hDevice 设备对象句柄，它应由设备的CreateDevice创建。
bDIOSts[8] bDIOSts 参数(注意: 必须定义为 8 个字节元素的数组)。
返回值：若成功返回 FALSE，否则返回 TRUE。
相关函数: CreateDevice EnableStsP1DIO SetDeviceDO_P1A GetDeviceDI_P1A SetDeviceDO_P1B GetDeviceDI_P1B SetDeviceDO_P1C GetDeviceDI_P1C ReleaseDevice

♦开关量输入状态

函数原型：

Visual C++：

```
BOOL GetDeviceDI_P1C (HANDLE hDevice,
                      BYTE bDIOSts[8])
```

Visual Basic:

```
Declare Function PCIE2321_GetDeviceDI_P1C Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                    ByRef bDIOsts As Byte) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：开关量输入状态。
参数：
hDevice 设备对象句柄，它应由设备的CreateDevice创建。
bDIOSts[8] bDIOSts 参数(注意: 必须定义为 8 个字节元素的数组)。
返回值：若成功返回 FALSE，否则返回 TRUE。
相关函数: CreateDevice EnableStsP1DIO SetDeviceDO_P1A GetDeviceDI_P1A SetDeviceDO_P1B GetDeviceDI_P1B SetDeviceDO_P1C GetDeviceDI_P1C ReleaseDevice

对 P2 操作

♦ **设置开关量输入或输出状态**

函数原型：

Visual C++：

```
BOOL EnableStsP2DIO (HANDLE hDevice,
                     BOOL bExtTrig,
                     BOOL bP1Sts[3])
```

Visual Basic:

```
Declare Function PCIE2321_EnableStsP2DIO Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                            ByVal bExtTrig As Boolean,_
                                                            ByRef bP2Sts As Long) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：设置开关量输入或输出状态。

参数：

hDevice 设备对象句柄，它应由设备的CreateDevice创建。

bExtTrig 是否使能外部触发功能 TRUE：使能，FALSE：禁止。

bP1Sts[3] bP1Sts[0]bP1Sts[2]分别控制 P1A、P1B、P1C 端口 FALSE：开关量输入，TRUE：开关量输出

返回值：若成功返回 FALSE，否则返回 TRUE。

相关函数: CreateDevice EnableStsP2DIO SetDeviceDO_P2A GetDeviceDI_P2A SetDeviceDO_P2B GetDeviceDI_P2B SetDeviceDO_P2C GetDeviceDI_P2C ReleaseDevice

对 P2A 操作

♦**开关量输出状态**

函数原型：

Visual C++：

```
BOOL SetDeviceDO_P2A (HANDLE hDevice,
                      BYTE bDIOSts[8])
```

Visual Basic:

```
Declare Function PCIE2321_SetDeviceDO_P2A Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                             ByRef bDIOsts As Byte) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：开关量输出状态。

参数：

hDevice 设备对象句柄，它应由设备的CreateDevice创建。

bDIOSts[8] bDIOSts 参数(注意: 必须定义为 8 个字节元素的数组)。

返回值：若成功返回 FALSE，否则返回 TRUE。

相关函数: CreateDevice EnableStsP2DIO SetDeviceDO_P2A GetDeviceDI_P2A SetDeviceDO_P2B GetDeviceDI_P2B SetDeviceDO_P2C GetDeviceDI_P2C ReleaseDevice

♦**开关量输入状态**

函数原型：

Visual C++：

```
BOOL GetDeviceDI_P2A (HANDLE hDevice,
                      BYTE bDIOSts[8])
```

Visual Basic:

```
Declare Function PCIE2321_GetDeviceDI_P2A Lib "PCIE2321_32" (ByVal hDevice As Long,_
```

ByRef bDIOsts As Byte) As Boolean

LabVIEW:

请参考相关演示程序。

功能：开关量输入状态。

参数：

hDevice 设备对象句柄，它应由设备的CreateDevice创建。

bDIOSts[8] bDIOSts 参数(注意：必须定义为 8 个字节元素的数组)。

返回值：若成功返回 FALSE，否则返回 TRUE。

相关函数： CreateDevice EnableStsP2DIO SetDeviceDO_P2A GetDeviceDI_P2A SetDeviceDO_P2B GetDeviceDI_P2B SetDeviceDO_P2C GetDeviceDI_P2C ReleaseDevice

对 P2B 操作

♦开关量输出状态

函数原型：

Visual C++：

```
BOOL SetDeviceDO_P2B (HANDLE hDevice,
                      BYTE bDIOSts[8])
```

Visual Basic:

```
Declare Function PCIE2321_SetDeviceDO_P2B Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                             ByRef bDIOsts As Byte) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：开关量输出状态。

参数：

hDevice 设备对象句柄，它应由设备的CreateDevice创建。

bDIOSts[8] bDIOSts 参数(注意：必须定义为 8 个字节元素的数组)。

返回值：若成功返回 FALSE，否则返回 TRUE。

相关函数： CreateDevice EnableStsP2DIO SetDeviceDO_P2A GetDeviceDI_P2A SetDeviceDO_P2B GetDeviceDI_P2B SetDeviceDO_P2C GetDeviceDI_P2C ReleaseDevice

♦开关量输入状态

函数原型：

Visual C++：

```
BOOL GetDeviceDI_P2B (HANDLE hDevice,
                      BYTE bDIOSts[8])
```

Visual Basic:

```
Declare Function PCIE2321_GetDeviceDI_P2B Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                             ByRef bIDOsts As Byte) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：开关量输入状态。

参数：

hDevice 设备对象句柄，它应由设备的CreateDevice创建。

bDIOSts[8] bDIOSts 参数(注意：必须定义为 8 个字节元素的数组)。

返回值：若成功返回 FALSE，否则返回 TRUE。

相关函数： CreateDevice EnableStsP2DIO SetDeviceDO_P2A GetDeviceDI_P2A SetDeviceDO_P2B GetDeviceDI_P2B SetDeviceDO_P2C GetDeviceDI_P2C ReleaseDevice

对 P2C 操作

♦开关量输出状态

函数原型：

Visual C++：

```
BOOL SetDeviceDO_P2C (HANDLE hDevice,
                      BYTE bDIOSts [8])
```

Visual Basic:

```
Declare Function PCIE2321_SetDeviceDO_P2C Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                             ByRef bDIOsts As Byte) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：开关量输出状态。

参数：

hDevice 设备对象句柄，它应由设备的CreateDevice创建。

bDIOSts[8] bDIOSts 参数(注意: 必须定义为 8 个字节元素的数组)。

返回值：若成功返回 FALSE，否则返回 TRUE。

相关函数: CreateDevice EnableStsP2DIO SetDeviceDO_P2A GetDeviceDI_P2A SetDeviceDO_P2B GetDeviceDI_P2B SetDeviceDO_P2C GetDeviceDI_P2C ReleaseDevice

♦开关量输入状态

函数原型：

Visual C++：

```
BOOL GetDeviceDI_P2C (HANDLE hDevice,
                      BYTE bDIOSts[8])
```

Visual Basic:

```
Declare Function PCIE2321_GetDeviceDI_P2C Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                             ByRef bDIOsts As Byte) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：开关量输入状态。

参数：

hDevice 设备对象句柄，它应由设备的CreateDevice创建。

bDIOSts[8] bDIOSts 参数(注意: 必须定义为 8 个字节元素的数组)。

返回值：若成功返回 FALSE，否则返回 TRUE。

相关函数: CreateDevice EnableStsP2DIO SetDeviceDO_P2A GetDeviceDI_P2A SetDeviceDO_P2B GetDeviceDI_P2B SetDeviceDO_P2C GetDeviceDI_P2C ReleaseDevice

第四节、计数器控制函数

♦设置计数器初值

函数原型：

Visual C++:

```
BOOL SetDevCNTInitValue (HANDLE hDevice
                         ULONG InitCntrVal)
```

Visual Basic:

```
Declare Function PCIE2321_SetDevCNTInitValue Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                                ByVal InitCntrVal As Long) As Boolean
```

LabVIEW:

请参考相关演示程序。

功能：设置计数器初值。
参数：hDevice 设备对象句柄，它应由CreateDevice创建。
InitCntrVal 32 位计数初值
返回值：如果设置成功，则返回 TRUE，否则返回 FALSE。
相关函数： CreateDevice SetDevCNTInitValue EnableDevCNT
GetDevCNTVal ResetDevCNT ReleaseDevice

♦ 是否使能计数器

函数原型：
Visual C++:

```
BOOL EnableDevCNT (HANDLE hDevice
                   BOOL bEnalbe)
```

Visual Basic:

```
Declare Function PCIE2321_EnableDevCNT Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                          ByVal bEnalbe As Boolean) As Boolean
```

LabVIEW:
请参考相关演示程序。

功能：是否使能计数器。
参数：hDevice 设备对象句柄，它应由CreateDevice创建。
bEnalbe TRUE：使能计数器 FALSE：禁止计数器
返回值：如果调用成功，则返回 TRUE，否则返回 FALSE。
相关函数： CreateDevice SetDevCNTInitValue EnableDevCNT
GetDevCNTVal ResetDevCNT ReleaseDevice

♦ 取得所有计数器的计数值

函数原型：
Visual C++:

```
BOOL GetDevCNTVal (HANDLE hDevice
                   PULONG pCntrValue)
```

Visual Basic:

```
Declare Function PCIE2321_GetDevCNTVal Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                          ByRef pCntrValue As Long) As Boolean
```

LabVIEW:
请参考相关演示程序。

功能：取得所有计数器的计数值。
参数：hDevice 设备对象句柄，它应由CreateDevice创建。
pCntrValue 计数值
返回值：如果调用成功，则返回 TRUE，否则返回 FALSE。
相关函数： CreateDevice SetDevCNTInitValue EnableDevCNT
GetDevCNTVal ResetDevCNT ReleaseDevice

♦ 计数器复位清零

函数原型：
Visual C++:

```
BOOL ResetDevCNT (HANDLE hDevice)
```

Visual Basic:

```
Declare Function PCIE2321_ResetDevCNT Lib "PCIE2321_32" (ByVal hDevice As Long) As Boolean
```

LabVIEW:
请参考相关演示程序。

功能：计数器复位清零。
参数：hDevice 设备对象句柄，它应由CreateDevice创建。
返回值：如果调用成功，则返回 TRUE，否则返回 FALSE。
相关函数: CreateDevice SetDevCNTInitValue EnableDevCNT GetDevCNTVal ResetDevCNT ReleaseDevice

第五节、中断函数

- **初始化中断**

函数原型：
Visual C++:

```
BOOL InitDeviceInt (HANDLE hDevice,
                    HANDLE hEventInt,
                    LONG INTSelect,
                    LONG INTSource)
```

Visual Basic:

```
Declare Function PCIE2321_InitDeviceInt Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                           ByVal hEventInt As Long,_
                                                           ByVal INTSelect As Long,_
                                                           ByVal INTSource As Long) As Boolean
```

LabVIEW:
请参考相关演示程序。
功能：初始化中断。
参数：
hDevice 设备对象句柄，它应由CreateDevice创建。
hEventInt 中断事件句柄,它由 CreateSystemEvent 创建。
INTSelect 选择 INT1 或 INT2 为总中断源。
INTSource INT1 或 INT2 的中断源选择。
返回值：如果成功，则返回 TRUE，否则返回 FALSE。
相关函数: CreateDevice SetDevFreqencyAD GetDeviceIntCount ReleaseDeviceInt ReleaseDevice

- **在中断初始化后，用它取得中断服务程序产生的次数**

函数原型：
Visual C++:

```
LONG GetDeviceIntCount (HANDLE hDevice)
```

Visual Basic:

```
Declare Function PCIE2321_GetDeviceIntCount Lib "PCIE2321_32" (ByVal hDevice As Long) As Long
```

LabVIEW:
请参考相关演示程序。

功能：在中断初始化后，用它取得中断服务程序产生的次数。
参数：
hDevice 设备对象句柄，它应由CreateDevice创建。
返回值：如果成功，则返回 TRUE，否则返回 FALSE。
相关函数: CreateDevice SetDevFreqencyAD GetDeviceIntCount ReleaseDeviceInt ReleaseDevice

- **释放中断资源**

函数原型：
Visual C++:

```
BOOL ReleaseDeviceInt (HANDLE hDevice)
```

Visual Basic:

Declare Function PCIE2321_ReleaseDeviceInt Lib "PCIE2321_32" (ByVal hDevice As Long) As Long

LabVIEW:

请参考相关演示程序。

功能：释放中断资源。

参数：

hDevice 设备对象句柄，它应由CreateDevice创建。

返回值：如果成功，则返回 TRUE，否则返回 FALSE。

相关函数: CreateDevice SetDevFreqencyAD GetDeviceIntCount ReleaseDeviceInt ReleaseDevice

第四章 硬件参数结构

第一节、硬件参数说明

InitDeviceInt 中的 INTSelect 使用的中断选择选项。

常量名	常量值	功能定义
PCIE2321_SELECT_INT1	0x00	总中断源为 INT1
PCIE2321_SELECT_INT2	0x01	总中断源为 INT2

InitDeviceInt 中的 INTSelect 选择 INT1 作为总中断源时 INTSource 所使用的选项。

常量名	常量值	功能定义
PCIE2321_INT1SRC_P1C0	0x00	P1C0 下降沿产生中断
PCIE2321_INT1SRC_P1C0P1C3	0x01	P1C0 与 P1C3 口产生中断源
PCIE2321_INT1SRC_EXCNT	0x02	外部计数器中断源

InitDeviceInt 中的 INTSelect 选择 INT2 作为总中断源时 INTSource 所使用的选项。

常量名	常量值	功能定义
PCIE2321_INT2SRC_P1C2P2C3	0x00	P1C2 或 P2C3 口产生中断
PCIE2321_INT2SRC_P2C0	0x01	P2C0 下降沿产生中断
PCIE2321_INT2SRC_INCNT	0x02	内部定时器中断源

第五章 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GetDeviceDI函数进行更便捷式数字量输入操作

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[PCIe2321 48 路 DIO 卡]丨[Microsoft Visual C++]丨[简易代码演示]丨[DIO...]

二、怎样使用SetDeviceDO函数进行更便捷式数字量输出操作

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[PCIe2321 48 路 DIO 卡]丨[Microsoft Visual C++]丨[简易代码演示]丨[DIO...]

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 PCIe2321.h)。

[程序]丨[阿尔泰测控演示系统]丨[PCIe2321 48 路 DIO 卡]丨[Microsoft Visual C++]丨[高级代码演示]

其默认存放路径为：系统盘\ART\PCIe2321\SAMPLES\VC\ADVANCED

其他语言的演示可以用上面类似的方法找到。

第六章 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“PCIe2321_”）

函数名	函数功能	备注
① PCIe 总线内存映射寄存器操作函数		
GetDeviceBar	取得指定的设备寄存器组 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 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	

第二节、PCIe 内存映射寄存器操作函数原型说明

◆ **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,
                   __int64 pbPCIBar[6])
```

Visual Basic:

```
Declare Function PCIE2321_GetDeviceBar Lib "PCIE2321_32" (ByVal hDevice As Long,_
                                                          ByRef pulPCIBar 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。

pbPCIBar[6] 返回 PCI BAR 所有地址,具体 PCI BAR 中有多少可用地址请看硬件说明书。

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 PCIE2321_GetDevVersion Lib "PCIE2321_32" (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 ReleaseDevice

- **以单字节（即 8 位）方式写 PCIe 内存映射寄存器的某个单元**

函数原型:

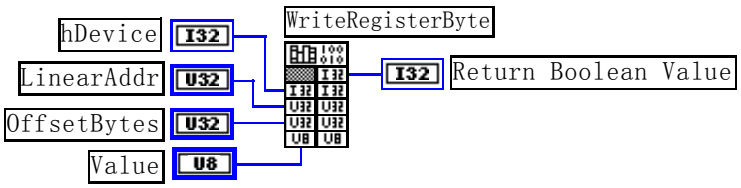
Visual C++:

BOOL WritePortByte (HANDLE hDevice,
__int64 pbPort,
BYTE Value)

Visual Basic:

Declare Function PCIE2321_WritePortByte Lib "PCIE2321_32" (ByVal hDevice As Long, _
ByVal pbPort As Long, _
ByVal Value As Byte) As Boolean

LabVIEW:



功能：以单字节（即 8 位）方式写 PCIe 内存映射寄存器。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

pbPort 指定寄存器的物理基地址。

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 PCIe 内存映射寄存器的某个单元**

函数原型:

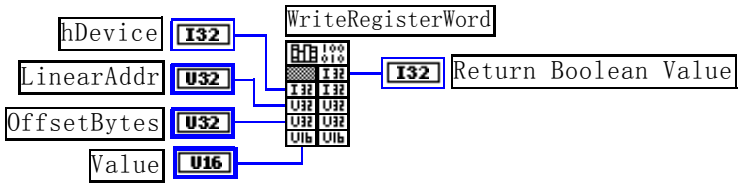
Visual C++:

```
BOOL WritePortWord (HANDLE hDevice,
                    __int64 pbPort,
                    WORD Value)
```

Visual Basic:

```
Declare Function PCIE2321_WritePortWord Lib "PCIE2321_32" (ByVal hDevice As Long, _
                                                           ByVal pbPort As Long, _
                                                           ByVal Value As Integer) As Boolean
```

LabVIEW:



功能：以双字节（即 16 位）方式写 PCIe 内存映射寄存器。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

pbPort 指定寄存器的物理基地址。

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 PCIe 内存映射寄存器的某个单元**

函数原型:

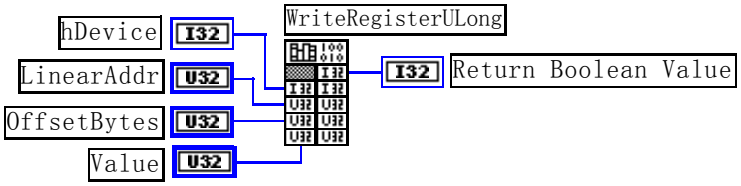
Visual C++:

```
BOOL WritePortULong (HANDLE hDevice,
                     __int64 pbPort,
                     ULONG Value)
```

Visual Basic:

```
Declare Function PCIE2321_WritePortULong Lib "PCIE2321_32" (ByVal hDevice As Long, _
                                                            ByVal pbPort As Long, _
                                                            ByVal Value As Long) As Boolean
```

LabVIEW:



功能：以四字节（即 32 位）方式写 PCIe 内存映射寄存器。

参数：

hDevice设备对象句柄，它应由CreateDevice创建。

pbPort 指定寄存器的物理基地址。

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 PCIe 内存映射寄存器的某个单元**

函数原型:

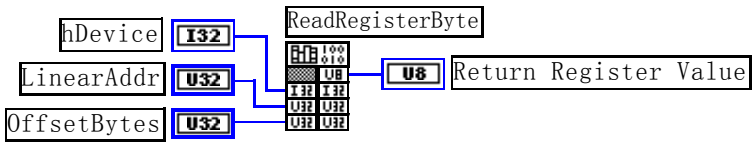
Visual C++:

```
BYTE ReadPortByte (HANDLE hDevice,
                   __int64 pbPort)
```

Visual Basic:

```
Declare Function PCIE2321_ReadPortByte Lib "PCIE2321_32" (ByVal hDevice As Long, _
                                                          ByVal pbPort As Long) As Byte
```

LabVIEW:



功能：以单字节（即 8 位）方式读 PCIe 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。

pbPort 指定寄存器的物理基地址。

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 PCIe 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 PCIe 内存映射寄存器的某个单元**

函数原型:

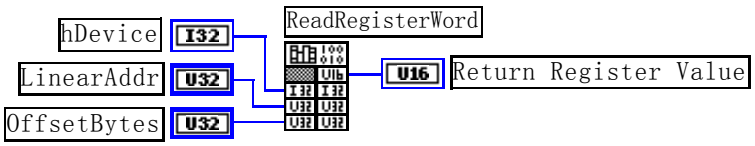
Visual C++:

```
WORD ReadPortWord (HANDLE hDevice,
                   __int64 pbPort)
```

Visual Basic:

```
Declare Function PCIE2321_ReadPortWord Lib "PCIE2321_32" (ByVal hDevice As Long, _
                                                          ByVal pbPort As Long) As Integer
```

LabVIEW:



功能：以双字节（即 16 位）方式读 PCIe 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。

pbPort 指定寄存器的物理基地址。

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 PCIe 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 PCIe 内存映射寄存器的某个单元**

函数原型:

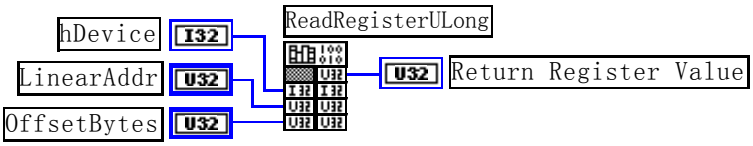
Visual C++:

```
ULONG ReadPortULong (HANDLE hDevice,
                     __int64 pbPort)
```

Visual Basic:

```
Declare Function PCIE2321_ReadPortULong Lib "PCIE2321_32" (ByVal hDevice As Long, _
                                                           ByVal pbPort As Long) As Long
```

LabVIEW:



功能：以四字节（即 32 位）方式读 PCIe 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。

pbPort 指定寄存器的物理基地址。

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 PCIe 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、线程操作函数原型说明

♦ **创建内核系统事件**

函数原型：

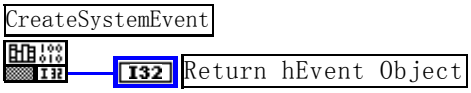
Visual C++:

HANDLE CreateSystemEvent(void)

Visual Basic:

Declare Function PCIE2321_CreateSystemEvent Lib "PCIE2321_32" () As Long

LabVIEW:



功能：创建系统内核事件对象，供 InitDeviceInt 和 VB 子线程等函数使用。

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 PCIE2321_ReleaseSystemEvent Lib "PCIE2321_32" (ByVal hEvent As Long) As Boolean

LabVIEW:

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。