ART2535
数字量输入输出卡

WIN2000/XP 驱动程序使用说明书

北京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 2
第一节、版权信息 2
第二节、命名约定 2
第二章　使用纲要 2
第一节、如何管理 PC104 设备 2
第二节、如何实现开关量的简便操作 2
第三节、哪些函数对您不是必须的 2
第三章　PC104 设备专用函数接口介绍 3
第一节、设备驱动接口函数列表（每个函数省略了前缀“ART2535_”） 3
第二节、设备对象管理函数原型说明 4
第三节、DIO 数字开关量输入输出简易操作函数原型说明 5
第四章　上层用户函数接口应用实例 12
第一节、简易程序演示说明 12
第二节、高级程序演示说明 13
第五章　公共接口函数介绍 13
第一节、公用接口函数总列表（每个函数省略了前缀“ART2535_”） 13
第二节、IO 端口读写函数原型说明 13

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 ART2535Inst.doc 文档。

第一章 版权信息与命名约定

第一节、版权信息

本软件产品及相关套件均属北京市阿尔泰科贸有限公司所有，其产权受国家法律绝对保护，除非本公司书面允许，其他公司、单位及个人不得非法使用和拷贝，否则将受到国家法律的严厉制裁。您若需要我公司产品及相关信息请及时与我们联系，我们将热情接待。

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 ARTxxxx_ 则被省略。如 ART2535_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、如何管理 PC104 设备

由于我们的驱动程序采用面向对象编程，所以要使用设备的一切功能，则必须首先用CreateDevice函数创建一个设备对象句柄hDevice，有了这个句柄，您就拥有了对该设备的控制权。然后将此句柄作为参数传递给其他函数，如SetDeviceDO函数可用实现开关量的输出等。最后可以通过ReleaseDevice将hDevice释放掉。

第二节、如何实现开关量的简便操作

当您有了hDevice设备对象句柄后，便可用SetDeviceDO函数实现开关量的输出操作，其各路开关量的输出状态由其bDOISts[8]中的相应元素决定。

第三节、哪些函数对您不是必须的

当公共函数如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。

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。

第三章　PC104 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“ART2535_”）

本章函数是设备使用 PC104 方式传输时所使用的。

函数名	函数功能	备注
① 设备对象操作函数		
CreateDevice	创建 PC104 对象(用设备逻辑号)	
ReleaseDevice	关闭设备，且释放 PC104 总线设备对象	
② 开关量函数		
EnableStsDIO	设置开关量输入或输出状态	上层用户
GetDeviceDI_PA	DIO0～DIO7 开关输入函数	上层用户
SetDeviceDO_PA	DIO0～DIO7 开关输出函数	上层用户
GetDeviceDI_PB	DIO8～DIO15 开关输入函数	上层用户
SetDeviceDO_PB	DIO8～DIO15 开关输出函数	上层用户
GetDeviceDI_PC	DIO16～DIO23 开关输入函数	上层用户
SetDeviceDO_PC	DIO16～DIO23 开关输出函数	上层用户
GetDeviceDI_PD	DIO24～DIO31 开关输入函数	上层用户
SetDeviceDO_PD	DIO24～DIO31 开关输出函数	上层用户
GetDeviceDI_PE	DIO32～DIO39 开关输入函数	上层用户
SetDeviceDO_PE	DIO32～DIO39 开关输出函数	上层用户
GetDeviceDI_PF	DIO40～DIO47 开关输入函数	上层用户
SetDeviceDO_PF	DIO40～DIO47 开关输出函数	上层用户

使用需知

Visual C++ & C++Builder：

首先将 ART2535.h 和 ART2535.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(ART2535.lib)加入到您的工程中。

#include “ART2535.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 自身的各种函数，其方法一样简单，毫无二别。

关于 ART2535.h 和 ART2535.lib 两个文件均可在演示程序文件夹下面找到。

Visual Basic:

首先将 ART2535.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 ART2535.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 ART2535.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 ART2535.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：“ART2535”。如：

```
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ART2535;  // 注意： 在此加入驱动程序接口单元 ART2535
```

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：

```
HANDLE CreateDevice(WORD wBaseAddress)
```

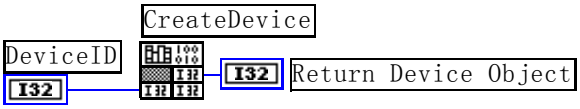
Visual Basic：

```
Declare Function CreateDevice Lib "ART2535" (ByVal wBaseAddress As Integer) As Long
```

Delphi：

```
Function CreateDevice(wBaseAddress : Word):Integer;
                    StdCall; External 'ART2535' 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：

wBaseAddress 设备基地址。板基地址可设置成 200H～3F0H 之间可被 16 整除的二进制码，板基地址默认为 300H，将占用基地址起的连续 32 个 I/O 地址。具体设置请参考硬件说明书。

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：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:

BOOL ReleaseDevice(HANDLE hDevice)

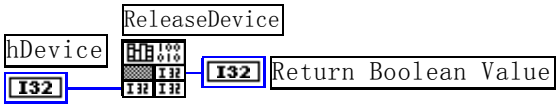
Visual Basic:

Declare Function ReleaseDevice Lib "ART2535" (ByVal hDevice As Long) As Boolean

Delphi:

Function ReleaseDevice(hDevice : Integer):Boolean;
StdCall; External 'ART2535' 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， 用户可以用 GetLastError 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，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。

第三节、DIO 数字开关量输入输出简易操作函数原型说明

♦ **设置开关量输入或输出状态**

函数原型：

Visual C++ & C++Builder:

BOOL EnableStsDIO(HANDLE hDevice,
BOOL bSts[6])

Visual Basic:

Declare Function EnableStsDIO Lib "ART2535" (ByVal hDevice As Long, _
ByVal bSts(0 to 5) As Boolean) As Boolean

Delphi:

Function EnableStsDIO (hDevice : Integer;
bSts: Pointer):Boolean;
StdCall; External 'ART2535' Name ' EnableStsDIO ';

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CreateDevice或CreateDeviceEx创建。

bSts bSts[0]控制 PA 端口（DIO0～DIO7），bSts[1]控制 PB 端口（DIO8～DIO15），bSts[2]控制 PC 端口（DIO16～DIO23），bSts[3]控制 PD 端口（DIO24～DIO31），bSts[4]控制 PE 端口（DIO32～DIO39），bSts[5]控制 PF 端口（DIO40～DIO47）。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数： CreateDevice ReleaseDevice

一、对PA口操作

♦ **PA 口 DIO0～DIO7 八路开关量输入**

函数原型：

Visual C++ & C++Builder:

```
BOOL GetDeviceDI_PA (HANDLE hDevice,
                     BYTE bDOISts [8])
```

Visual Basic:

```
Declare Function GetDeviceDI_PA Lib "ART2535" (ByVal hDevice As Long, _
                                               ByVal bDOISts (0 to 7 ) As Byte) As Boolean
```

Delphi:

```
Function GetDeviceDI_PA ( hDevice : Integer;
                          bDOISts : Pointer):Boolean;
                          StdCall; External 'ART2535' Name ' GetDeviceDI_PA ';
```

LabVIEW:

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO0～DIO7 输入开关量状态读入内存。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。

bDOISts八路开关量输入状态的参数结构，共有 8 个成员变量，分别对应于DIO0～DIO7 路开关量输入状态位。如果bDOISts [0]为“1”则使 0 通道处于开状态，若为“0”则 0 通道为关状态。其他同理。具体定义请参考《DI数字开关量输入参数介绍》章节。

返回值：若成功，返回 TRUE，其 bDOISts[x]中的值有效；否则返回 FALSE，其 bDOISts[x]中的值无效。

相关函数： CreateDevice SetDeviceDO_PA ReleaseDevice

♦ PA 口 DIO0～DIO7 八路开关量输出

函数原型：

Visual C++ & C++Builder:

```
BOOL SetDeviceDO_PA (HANDLE hDevice,
                     BYTE bDOISts[8])
```

Visual Basic:

```
Declare Function SetDeviceDO_PA Lib "ART2535" ( ByVal hDevice As Long, _
                                                ByVal bDOISts(0 to 7 )As Byte) As Boolean
```

Delphi:

```
Function SetDeviceDO_PA (hDevice : Integer;
                         bDOISts : Pointer):Boolean;
                         1StdCall; External 'ART2535' Name ' SetDeviceDO_PA ';
```

LabView：

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO0～DIO7 输出开关量置成相应的状态。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。

bDOISts 八路开关量输出状态的参数结构，共有 8 个成员变量，分别对应于DIO0～DIO7 路开关量输出状态位。比如置bDOISts[0]为“1”则使 0 通道处于“开”状态，若为“0”则置 0 通道为“关”状态。其他同理。请注意，在实际执行这个函数之前，必须对这个参数结构的DIO0 至DIO7 共 8 个成员变量赋初值，其值必须为“1”或“0”。具体定义请参考《DO数字开关量输出参数介绍》。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数： CreateDevice GetDeviceDI_PA ReleaseDevice

二、对PB口操作

♦ PB 口 DIO8～DIO15 八路开关量输入

函数原型：

Visual C++ & C++Builder:

```
BOOL GetDeviceDI_PB (HANDLE hDevice,
                     BYTE bDOISts[8])
```

Visual Basic:

```
Declare Function GetDeviceDI_PB Lib "ART2535" (ByVal hDevice As Long, _
                                               ByVal bDOISts(0 to 7 ) As Byte) As Boolean
```

Delphi:

```
Function GetDeviceDI_PB ( hDevice : Integer;
                          bDOISts : Pointer):Boolean;
                          StdCall; External 'ART2535' Name ' GetDeviceDI_PB ';
```

LabVIEW:

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO8～DIO15 输入开关量状态读入内存。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。bDOISts八路开关量输入状态的参数结构，共有 8 个成员变量，分别对应于DIO8～DIO15 路开关量输入状态位。如果bDOISts[0]为“1”则使 0 通道处于开状态，若为“0”则 0 通道为关状态。其他同理。具体定义请参考《DI数字开关量输入参数介绍》章节。

返回值：若成功，返回 TRUE，其 bDOISts[x]中的值有效；否则返回 FALSE，其 bDOISts[x]中的值无效。

相关函数： CreateDevice SetDeviceDO_PB ReleaseDevice

♦ PB 口 DIO8～DIO15 八路开关量输出

函数原型：

Visual C++ & C++Builder:

```
BOOL SetDeviceDO_PB (HANDLE hDevice,
                     BYTE bDOISts[8])
```

Visual Basic:

```
Declare Function SetDeviceDO_PB Lib "ART2535" ( ByVal hDevice As Long, _
                                                ByVal bDOISts(0 to 7 )As Byte) As Boolean
```

Delphi:

```
Function SetDeviceDO_PB (hDevice : Integer;
                         bDOISts : Pointer):Boolean;
                         1StdCall; External 'ART2535' Name ' SetDeviceDO_PB ';
```

LabView：

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO8～DIO15 输出开关量置成相应的状态。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。

bDOISts 八路开关量输出状态的参数结构，共有 8 个成员变量，分别对应于DIO8～DIO15 路开关量输出状

态位。比如置bDOISts[0]为“1”则使 0 通道处于“开”状态，若为“0”则置 0 通道为“关”状态。其他同理。请注意，在实际执行这个函数之前，必须对这个参数结构的DIO8 至DIO15 共 8 个成员变量赋初值，其值必须为“1”或“0”。具体定义请参考《DO数字开关量输出参数介绍》。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数： CreateDevice GetDeviceDI_PB ReleaseDevice

三、对PC口操作

♦ PC 口 DIO16～DIO23 八路开关量输入

函数原型：

Visual C++ & C++Builder:

```
BOOL GetDeviceDI_PC (HANDLE hDevice,
                     BYTE bDOISts[8])
```

Visual Basic:

```
Declare Function GetDeviceDI_PC Lib "ART2535" (ByVal hDevice As Long, _
                                               ByVal bDOISts(0 to 7 ) As Byte) As Boolean
```

Delphi:

```
Function GetDeviceDI_PC ( hDevice : Integer;
                          bDOISts : Pointer):Boolean;
                          StdCall; External 'ART2535' Name ' GetDeviceDI_PC ';
```

LabVIEW:

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO16～DIO23 输入开关量状态读入内存。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。bDOISts八路开关量输入状态的参数结构，共有 8 个成员变量，分别对应于DIO16～DIO23 路开关量输入状态位。如果bDOISts[0]为“1”则使 0 通道处于开状态，若为“0”则 0 通道为关状态。其他同理。具体定义请参考《DI数字开关量输入参数介绍》章节。

返回值：若成功，返回 TRUE，其 bDOISts[x]中的值有效；否则返回 FALSE，其 bDOISts[x]中的值无效。

相关函数： CreateDevice SetDeviceDO_PC ReleaseDevice

♦ PC 口 DIO16～DIO23 八路开关量输出

函数原型：

Visual C++ & C++Builder:

```
BOOL SetDeviceDO_PC (HANDLE hDevice,
                     BYTE bDOISts[8])
```

Visual Basic:

```
Declare Function SetDeviceDO_PC Lib "ART2535" ( ByVal hDevice As Long, _
                                                ByVal bDOISts(0 to 7 )As Byte) As Boolean
```

Delphi:

```
Function SetDeviceDO_PC (hDevice : Integer;
                         bDOISts : Pointer):Boolean;
                         1StdCall; External 'ART2535' Name ' SetDeviceDO_PC ';
```

LabView：

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO16～DIO23 输出开关量置成相应的状态。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。

bDOISts 八路开关量输出状态的参数结构，共有 8 个成员变量，分别对应于DIO16～DIO23 路开关量输出状态位。比如置bDOISts[0]为“1”则使 0 通道处于“开”状态，若为“0”则置 0 通道为“关”状态。其他同理。请注意，在实际执行这个函数之前，必须对这个参数结构的DIO16 至DIO23 共 8 个成员变量赋初值，其值必须为“1”或“0”。具体定义请参考《DO数字开关量输出参数介绍》。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数： CreateDevice GetDeviceDI_PC ReleaseDevice

四、对PD口操作

♦ PD 口 DIO24～DIO31 八路开关量输入

函数原型：

Visual C++ & C++Builder:

```
BOOL GetDeviceDI_PD (HANDLE hDevice,
                     BYTE bDOISts [8])
```

Visual Basic:

```
Declare Function GetDeviceDI_PD Lib "ART2535" (ByVal hDevice As Long, _
                                               ByVal bDOISts (0 to 7 ) As Byte) As Boolean
```

Delphi:

```
Function GetDeviceDI_PD ( hDevice : Integer;
                          bDOISts : Pointer):Boolean;
                          StdCall; External 'ART2535' Name ' GetDeviceDI_PD ';
```

LabVIEW:

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO24～DIO31 输入开关量状态读入内存。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。

bDOISts八路开关量输入状态的参数结构，共有 8 个成员变量，分别对应于DIO24～DIO31 路开关量输入状态位。如果bDOISts [0]为“1”则使 0 通道处于开状态，若为“0”则 0 通道为关状态。其他同理。具体定义请参考《DI数字开关量输入参数介绍》章节。

返回值：若成功，返回 TRUE，其 bDOISts[x]中的值有效；否则返回 FALSE，其 bDOISts[x]中的值无效。

相关函数： CreateDevice SetDeviceDO_PD ReleaseDevice

♦ PD 口 DIO24～DIO31 八路开关量输出

函数原型：

Visual C++ & C++Builder:

```
BOOL SetDeviceDO_PD (HANDLE hDevice,
                     BYTE bDOISts[8])
```

Visual Basic:

```
Declare Function SetDeviceDO_PD Lib "ART2535" ( ByVal hDevice As Long, _
                                                ByVal bDOISts(0 to 7 )As Byte) As Boolean
```

Delphi:

```
Function SetDeviceDO_PD (hDevice : Integer;
                         bDOISts : Pointer):Boolean;
```

1StdCall; External 'ART2535' Name ' SetDeviceDO_PD ';

LabView：

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO24～DIO31 输出开关量置成相应的状态。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。

bDOISts 八路开关量输出状态的参数结构，共有 8 个成员变量，分别对应于DIO24～DIO31 路开关量输出状态位。比如置bDOISts[0]为“1”则使 0 通道处于“开”状态，若为“0”则置 0 通道为“关”状态。其他同理。请注意，在实际执行这个函数之前，必须对这个参数结构的DIO24 至DIO31 共 8 个成员变量赋初值，其值必须为“1”或“0”。具体定义请参考《DO数字开关量输出参数介绍》。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数： CreateDevice GetDeviceDI_PD ReleaseDevice

五、对PE口操作

♦ PE 口 DIO32～DIO39 八路开关量输入

函数原型：

Visual C++ & C++Builder:

```
BOOL GetDeviceDI_PE (HANDLE hDevice,
                     BYTE bDOISts[8])
```

Visual Basic:

```
Declare Function GetDeviceDI_PE Lib "ART2535" (ByVal hDevice As Long, _
                                               ByVal bDOISts(0 to 7 ) As Byte) As Boolean
```

Delphi:

```
Function GetDeviceDI_PE ( hDevice : Integer;
                          bDOISts : Pointer):Boolean;
                          StdCall; External 'ART2535' Name ' GetDeviceDI_PE ';
```

LabVIEW:

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO32～DIO39 输入开关量状态读入内存。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。bDOISts八路开关量输入状态的参数结构，共有 8 个成员变量，分别对应于DIO32～DIO39 路开关量输入状态位。如果bDOISts[0]为“1”则使 0 通道处于开状态，若为“0”则 0 通道为关状态。其他同理。具体定义请参考《DI数字开关量输入参数介绍》章节。

返回值：若成功，返回 TRUE，其 bDOISts[x]中的值有效；否则返回 FALSE，其 bDOISts[x]中的值无效。

相关函数： CreateDevice SetDeviceDO_PE ReleaseDevice

♦ PE 口 DIO32～DIO39 八路开关量输出

函数原型：

Visual C++ & C++Builder:

```
BOOL SetDeviceDO_PE (HANDLE hDevice,
                     BYTE bDOISts[8])
```

Visual Basic:

```
Declare Function SetDeviceDO_PE Lib "ART2535" ( ByVal hDevice As Long, _
```

ByVal bDOISts(0 to 7)As Byte) As Boolean

Delphi:

Function SetDeviceDO_PE (hDevice : Integer;
bDOISts : Pointer):Boolean;
1StdCall; External 'ART2535' Name ' SetDeviceDO_PE ';

LabView：

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO32～DIO39 输出开关量置成相应的状态。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。

bDOISts 八路开关量输出状态的参数结构，共有 8 个成员变量，分别对应于DIO32～DIO39 路开关量输出状态位。比如置bDOISts[0]为“1”则使 0 通道处于“开”状态，若为“0”则置 0 通道为“关”状态。其他同理。请注意，在实际执行这个函数之前，必须对这个参数结构的DIO32 至DIO39 共 8 个成员变量赋初值，其值必须为“1”或“0”。具体定义请参考《DO数字开关量输出参数介绍》。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数： CreateDevice GetDeviceDI_PE ReleaseDevice

六、对PF口操作

♦ PF 口 DIO40～DIO47 八路开关量输入

函数原型：

Visual C++ & C++Builder:

BOOL GetDeviceDI_PF (HANDLE hDevice,
BYTE bDOISts[8])

Visual Basic:

Declare Function GetDeviceDI_PF Lib "ART2535" (ByVal hDevice As Long, _
ByVal bDOISts(0 to 7) As Byte) As Boolean

Delphi:

Function GetDeviceDI_PF (hDevice : Integer;
bDOISts : Pointer):Boolean;
StdCall; External 'ART2535' Name ' GetDeviceDI_PF ';

LabVIEW:

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO40～DIO47 输入开关量状态读入内存。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。bDOISts八路开关量输入状态的参数结构，共有 8 个成员变量，分别对应于DIO40～DIO47 路开关量输入状态位。如果bDOISts[0]为“1”则使 0 通道处于开状态，若为“0”则 0 通道为关状态。其他同理。具体定义请参考《DI数字开关量输入参数介绍》章节。

返回值：若成功，返回 TRUE，其 bDOISts[x]中的值有效；否则返回 FALSE，其 bDOISts[x]中的值无效。

相关函数： CreateDevice SetDeviceDO_PF ReleaseDevice

♦ PF 口 DIO40～DIO47 八路开关量输出

函数原型：

Visual C++ & C++Builder:

```
BOOL SetDeviceDO_PF (HANDLE hDevice,
                     BYTE bDOISts[8])
```

Visual Basic:

```
Declare Function SetDeviceDO_PF Lib "ART2535" ( ByVal hDevice As Long, _
                                                ByVal bDOISts(0 to 7 )As Byte) As Boolean
```

Delphi:

```
Function SetDeviceDO_PF (hDevice : Integer;
                         bDOISts : Pointer):Boolean;
                         1StdCall; External 'ART2535' Name ' SetDeviceDO_PF ';
```

LabView：

请参考相关演示程序。

功能：负责将 PC104 设备上的 DIO40～DIO47 输出开关量置成相应的状态。

参数：

hDevice 设备对象句柄，它应由CreateDevice或CreateDeviceEx创建。

bDOISts 八路开关量输出状态的参数结构，共有 8 个成员变量，分别对应于DIO40～DIO47 路开关量输出状态位。比如置bDOISts[0]为“1”则使 0 通道处于“开”状态，若为“0”则置 0 通道为“关”状态。其他同理。请注意，在实际执行这个函数之前，必须对这个参数结构的DIO40 至DIO47 共 8 个成员变量赋初值，其值必须为“1”或“0”。具体定义请参考《DO数字开关量输出参数介绍》。

返回值：若成功，返回 TRUE，否则返回 FALSE。

相关函数： CreateDevice GetDeviceDI_PF ReleaseDevice

❖ **以上函数调用一般顺序**

① CreateDevice

② SetDeviceDO_PX（或 GetDeviceDI_PX，当然这两个函数也可同时进行）

③ ReleaseDevice

用户可以反复执行第②步，以进行数字 I/0 的输入输出。

第四章 上层用户函数接口应用实例

如果您想快速的了解驱动程序的使用方法和调用流程，以最短的时间建立自己的应用程序，那么我们强烈建议您参考相应的简易程序。此种程序属于工程级代码，可以直接打开不用作任何配置和代码修改即可编译通过，运行编译链接后的可执行程序，即可看到预期效果。

如果您想了解硬件的整体性能、精度、采样连续性等指标以及波形显示、数据存盘与分析、历史数据回放等功能，那么请参考高级演示程序。特别是许多不愿意编写任何程序代码的用户，您可以使用高级程序进行采集、显示、存盘等功能来满足您的要求。甚至可以用我们提供的专用转换程序将高级程序采集的存盘文件转换成相应格式，即可在 Excel、MatLab 第三方软件中分析数据（此类用户请最好选用通过 Visual C++制作的高级演示系统）。

第一节、简易程序演示说明

一、怎样使用 GetDeviceDI 函数进行开关量输入操作

其详细应用实例及正确代码请参考 Visual C++简易演示系统及源程序，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 ART2535.h 和 Sys.cpp)。

[程序]丨**[阿尔泰测控演示系统]**丨**[ART2535 48 路开关量输入输出卡]**丨**[Microsoft Visual C++]**丨**[**简易代码演示**]**丨**[DI 开关量演示源程序]**

其默认存放路径为：系统盘\ART\ART2535\SAMPLES\VC\SIMPLE\DI

二、怎样使用 SetDeviceDO 函数进行开关量输出操作

其详细应用实例及正确代码请参考 Visual C++简易演示系统及源程序，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 ART2535.h 和 Sys.cpp)。

[程序]丨**[阿尔泰测控演示系统]**丿**[ART2535 48 路开关量输入输出卡]**丨**[Microsoft Visual C++]**丿**[简易代码演示]**丿**[DO 开关量演示源程序]**

其默认存放路径为：系统盘\ART\ART2535\SAMPLES\VC\SIMPLE\DO

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 ART2535.h 和 Sys.cpp)。

[程序]丨**[阿尔泰测控演示系统]**丿**[ART2535 48 路开关量输入输出卡]**丨**[Microsoft Visual C++]**丿**[高级代码演示]**

其默认存放路径为：系统盘\ART\ART2535\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　公共接口函数介绍

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“ART2535_”）

函数名	函数功能	备注
① 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

第二节、IO 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++ & C++ Builder:

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

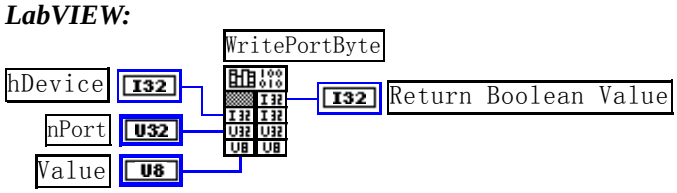
Visual Basic:

```
Declare Function WritePortByte Lib "ART2535" ( ByVal hDevice As Long, _
                                               ByVal nPort As Long, _
                                               ByVal Value As Byte) As Boolean
```

Delphi:

```
Function WritePortByte(hDevice : Integer;
                       nPort : LongWord;
                       Value : Byte) : Boolean;
                       StdCall; External 'ART2535' Name ' WritePortByte ';
```

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 设备的 I/O 端口号。

Value 写入由 nPort 指定端口的值。

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++ & C++ Builder:

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

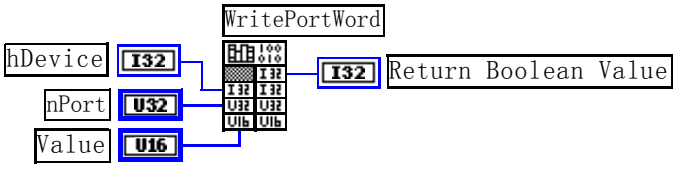
Visual Basic:

```
Declare Function WritePortWord Lib "ART2535" (ByVal hDevice As Long, _
                                              ByVal nPort As Long, _
                                              ByVal Value As Integer) As Boolean
```

Delphi:

```
Function WritePortWord(hDevice : Integer;
                       nPort : LongWord;
                       Value : Word) : Boolean;
                       StdCall; External 'ART2535' Name ' WritePortWord ';
```

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 设备的 I/O 端口号。

Value 写入由 nPort 指定端口的值。

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++ & C++ Builder:

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

Visual Basic:

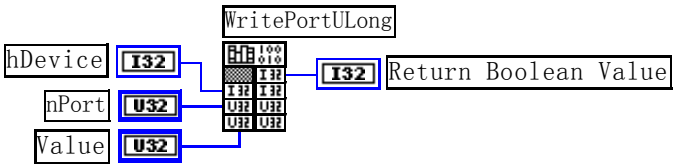
```
Declare Function WritePortULong Lib "ART2535" (ByVal hDevice As Long, _
                                               ByVal nPort As Long, _
```

ByVal Value As Long) As Boolean

Delphi:

```
Function WritePortULong(hDevice : Integer;
                        nPort : LongWord;
                        Value : LongWord) : Boolean;
                        StdCall; External 'ART2535' Name ' WritePortULong ';
```

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 设备的 I/O 端口号。

Value 写入由 nPort 指定端口的值。

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++ & C++ Builder:

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

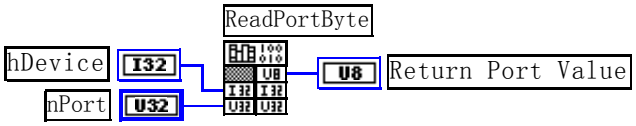
Visual Basic:

```
Declare Function ReadPortByte Lib "ART2535" (ByVal hDevice As Long, _
                                             ByVal nPort As Long ) As Byte
```

Delphi:

```
Function ReadPortByte(hDevice : Integer;
                      nPort : LongWord) : Byte;
                      StdCall; External 'ART2535' Name ' ReadPortByte ';
```

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 设备的 I/O 端口号。

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++ & C++ Builder:

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

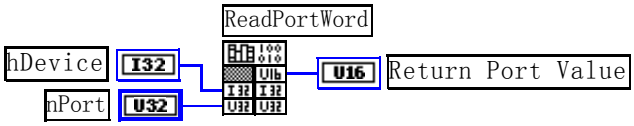
Visual Basic:

```
Declare Function ReadPortWord Lib "ART2535" ( ByVal hDevice As Long, _
                                              ByVal nPort As Long ) As Integer
```

Delphi:

```
Function ReadPortWord(hDevice : Integer;
                      nPort : LongWord) : Word;
                      StdCall; External 'ART2535' Name ' ReadPortWord ';
```

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　设备的 I/O 端口号。
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++ & C++ Builder:

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

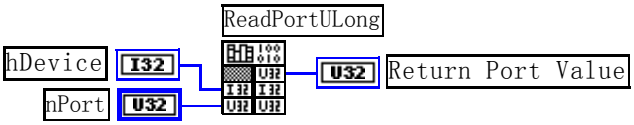
Visual Basic:

```
Declare Function ReadPortULong Lib "ART2535" ( ByVal hDevice As Long, _
                                               ByVal nPort As Long ) As Long
```

Delphi:

```
Function ReadPortULong(hDevice : Integer;
                       nPort : LongWord) : LongWord;
                       StdCall; External 'ART2535' Name ' ReadPortULong ';
```

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　设备的 I/O 端口号。
返回值：返回由 nPort 指定端口的值。
相关函数： CreateDevice　WritePortByte　WritePortWord
WritePortULong　ReadPortByte　ReadPortWord