

# ActiveDAS 控件

## 用户说明书

@版权2001 EVOC Technology Co., Ltd.  
版权所有

2001年04月20日第1版

为提高产品的可靠性,设计和功能,本文所有信息若有变更,恕不提前通知。本文信息也不作为厂商的任何承诺。任何情况下,包括已警告了的各种损坏的可能性,厂商均不负责直接的,非直接的,特殊的或偶然的因不正当使用本产品或文件所造成的损坏。

本文包含受版权保护的信息,版权所有。本手册的任何部分未经厂商书面同意,不得以机械的,电子的或其他任何方式复制。

### 商标

**ActiveDAS** 是 EVOC Technology Co., Ltd.的注册商标。Windows 95/98/NT, Visual Basic, Visual C++都是微软公司的注册商标。Intel 是 Intel 集团的注册商标。其他本文提及的产品名称在这里只作识别用途,可能是他们各自公司的商标或注册商标

# 目 录

目 录.....	1
<b>第一章 EVOC ACTIVE<span style="font-family: monospace;">DAS</span> 控件简介 .....</b>	<b>5</b>
1.1 什么是 EVOC ACTIVE <span style="font-family: monospace;">DAS</span> 控件.....	5
1.2 安装 EVOC ACTIVE <span style="font-family: monospace;">DAS</span> 控件 .....	6
1.2.1 系统需求.....	6
1.2.2 安装说明.....	6
1.2.3 安装文件.....	8
1.3 设置控件的属性 .....	8
1.3.1 属性页设置.....	8
1.3.2 编程设置.....	10
1.4 使用控件的方法 .....	10
1.5 使用控件的事件 .....	10
<b>第二章 在 VISUAL BASIC 中使用 EVOC ACTIVE<span style="font-family: monospace;">DAS</span> 控件 .....</b>	<b>11</b>
2.1 开发 VISUAL BASIC 应用程序.....	11
2.1.1 把 Evoc Active <span style="font-family: monospace;">DAS</span> 控件装入工具箱.....	11
2.1.2 用 Evoc Active <span style="font-family: monospace;">DAS</span> 控件建立用户界面 .....	12
2.1.3 在 Visual Basic 中使用属性页.....	12
2.1.4 用 Visual Basic 程序编辑属性.....	14
2.1.5 Visual Basic 中使用控件的方法.....	14
2.1.6 Visual Basic 中使用控件的事件.....	14
<b>第三章 在 VISUAL C++中使用 EVOC ACTIVE<span style="font-family: monospace;">DAS</span> 控件 .....</b>	<b>17</b>

---

3.1 开发 VISUAL C++应用程序.....	17
3.1.1 在 Visual C++ 中建立应用程序.....	17
3.1.2 把 Evoc ActiveDAS 控件装入工具箱.....	18
3.1.3 用 Evoc ActiveDAS 控件建立用户界面.....	18
3.1.4 用 Evoc ActiveDAS 控件编程.....	19
3.1.5 Visual C++ 中作用控件的属性.....	19
3.1.6 Visual C++ 中作用控件的方法.....	20
3.1.7 Visual C++ 中作用控件的事件.....	21
<b>第四章 在 DELPHI 中使用 EVOC ACTIVEVAS 控件.....</b>	<b>23</b>
4.1 开发 DELPHI 应用程序.....	23
4.1.1 把 Evoc ActiveDAS 控件装入组件面板.....	23
<b>第五章 EVOC ACTIVEVAS 控件的详细介绍.....</b>	<b>25</b>
5.1 E7SEGMENT 控件.....	25
5.2 BOOLEAN 控件.....	26
5.3 KNOB 控件.....	27
5.4 SLIDE 控件.....	30
5.5 MOTOR 控件.....	30
5.6 PIPE 控件.....	30
5.7 PUMP 控件.....	32
5.8 TANK 控件.....	32
5.9 VALUE 控件.....	33
5.10 GRAPH 控件.....	33
5.11 CHART 控件.....	38
5.12 DASDEVICE 控件.....	40
5.13 DASAI 控件.....	40
5.14 DASAO 控件.....	42
5.15 DASDI 控件.....	43

---

5.16	DASDO 控件.....	44
5.17	DASCOUNTER 控件.....	44
5.18	DASPULSE 控件 .....	45
5.19	DASALARM 控件 .....	45
5.20	PCL731A 控件.....	46
5.21	PCL733 控件.....	46
5.22	PCL734 控件.....	46
5.23	PCL836A 控件.....	47
<b>第六章</b>	<b>例子及注意事项.....</b>	<b>49</b>
6.1	DEMOUI10DIO 例子.....	49
6.2	DEMOUI10DA 例子 .....	51
6.3	DEMOUI10AD 例子 .....	53
6.4	GRAPHDEMO 例子 .....	56
6.5	CHARTDEMO 例子 .....	58
6.6	INSTRUMENT 综合例子.....	60
6.7	SYNTHESIZE 综合例子.....	63
6.8	VCGRAPHDEMO 例子.....	64
6.9	DASAI10DEMO 例子.....	64
6.10	DASAO10DEMO 例子.....	66
6.11	DASDI10DEMO 例子 .....	67
6.12	DASDO10DEMO 例子.....	68
6.13	其它注意事项.....	70
<b>第七章</b>	<b>硬件操作类控件支持的设备类型及其安装方法 .....</b>	<b>71</b>
7.1	各控件支持的本公司的卡的类型如下.....	71
7.2	安装方法 .....	71
<b>第八章</b>	<b>配置工具 DEVINST.EXE 的使用 .....</b>	<b>75</b>



## 第一章 Evoc ActiveDAS 控件简介

这一章总括Evoc ActiveDAS控件，列出了系统需求，描述了怎样安装软件，以及一些ActiveX控件的基本知识。

### 1.1 什么是 Evoc ActiveDAS 控件

Evoc ActivDAS 控件是关于工控领域的一些控件的集合，可用于任何支持 ActiveX 控件的容器程序。使用 Evoc ActiveDAS 控件，可以开发出复杂的用户界面应用程序，用于显示数据，控制数据的采集。Evoc ActiveDAS 控件包有以下三类：

- 1) 用户界面控件(EvocUI10.ocx) 32 位的控件用来采集数据，控制设备。包括 Boolean (布尔型)、E7Segment (七段数码管)、Knob (旋钮)、Slide (滚动条)。
- 2) 设备控件(EvocEquipment.ocx) 用来设计多媒体程序。包括 Motor (马达)、Pipe (管子)、Pump (泵)、Tank (罐)、Value (阀)。
- 3) 曲线控件(EvocChart10.ocx) Graph(历史曲线)、Chart (实时曲线)。
- 4) 硬件操作控件 主要用于本公司硬件的操作，方便实用，无需详细了解原来的 DLL，只需这些控件的使用方法即可。共包括八个控件和一个配置工具 (DevInst.exe)。有 DASDevice(设备)、DASAI (模拟量输入)、DASAO(模拟量输出)、DASDI(数字量输入)、DASDO(数字量输出)、DASCounter(计数器)、DASPulse (脉冲输出)、DASAlarm(报警)。

## 1.2 安装 Evoc ActiveDAS 控件

Evoc ActiveDAS安装程序的安装过程大约需要几分钟的时间。使用控件前要先注册MFC库，如：regsvr32 C:\windows\system\mfc42.dll。板卡控件使用前，先用本软件包中的配置工具 DevInst.exe 对卡的资源进行配置（包括端口、中断、DMA）。

### 1.2.1 系统需求

使用 Evoc ActiveDAS 控件，需要以下的系统需求：

- 1) Microsoft Windows95/98 或 Windows NT 操作系统。
- 2) PC 66MHz 80486 或更高配置。
- 3) VGA 分辨率适配卡或更高配置。
- 4) 建议屏幕设置在增强色(16 位)以上（切记），否认某些控件将显示失真。
- 5) ActiveX 控件容器，例如 Visual Basic（32 位版本）、Visual C++或 Delphi（32 位版本）。
- 6) 至少 16M 内存。
- 7) 硬盘剩余空间至少 10M 。
- 8) 微软兼容式鼠标。
- 9) 使用控件前要先注册 MFC 库，如：regsvr32 C:\windows\system\mfc42.dll
- 10) 板卡控件使用前,先用本软件包中的配置工具 DevInst.exe 对卡的资源进行配置（包括端口、中断、DMA）。具体配置方法参考第八章。

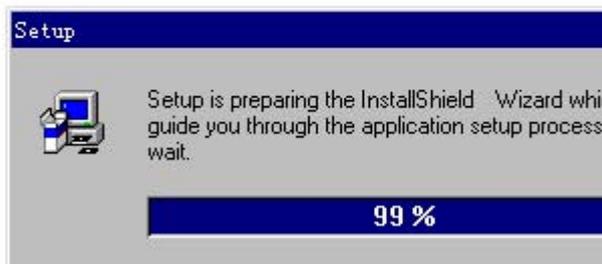
### 1.2.2 安装说明

这部分向您提供了怎样安装 Evoc ActiveDAS 控

件。

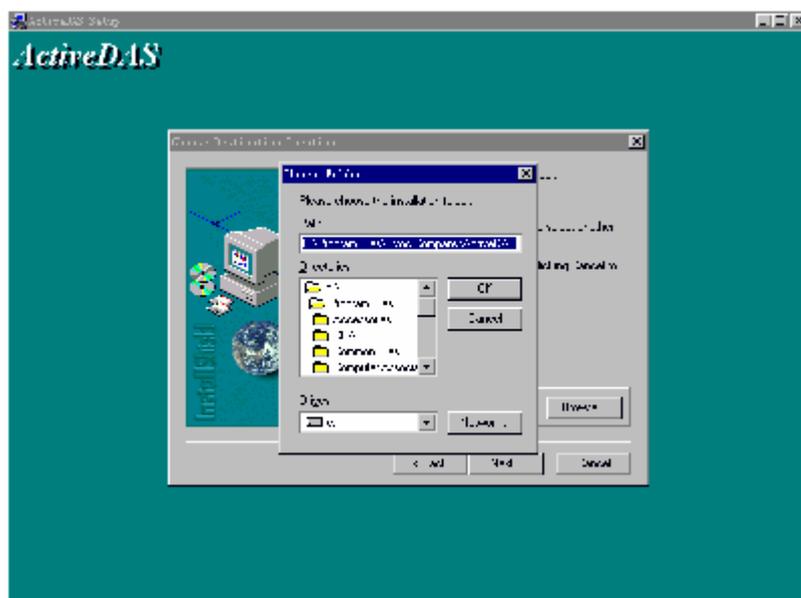
运行软盘或光盘的 Setup.exe 文件。根据屏幕的指示进行逐步安装即可。

当然，也可拷贝到硬盘上直接进行安装。开始看到的安装画面如下：



### 1.2.3 安装文件

安装过程中可改变文件的路径，如下图：



安装文件自动拷贝文件到\Windows\System 下并自动注册，安装文件还拷贝所有的例子程序到您所选择的目录下。

### 1.3 设置控件的属性

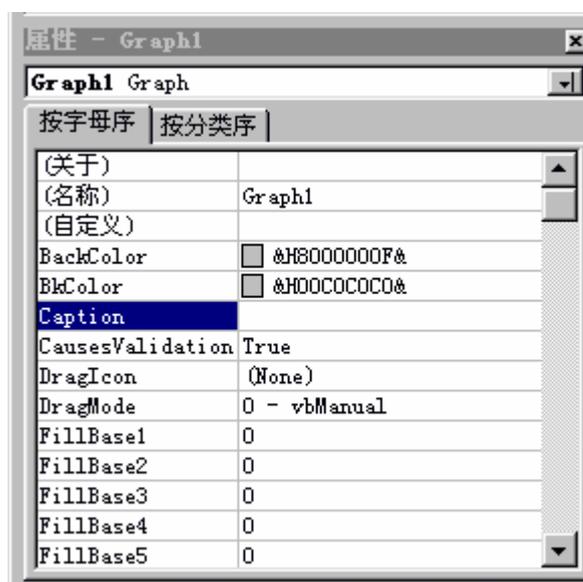
您可通过属性页或通过编程设置控件的属性。

#### 1.3.1 属性页设置

属性页是 Windows98 和 Windows NT 中常见的用户界面。当您改变对象的外观时，可用右键单击对象，选择属性项，会弹出相关的属性页。这时您可自定义控件的各种属性。

在有些编程环境中（如 Visual Basic 和 Delphi），

您有两种不同的属性页。对于编程环境的普通属性页叫做缺省属性，它包括控件的大部分基本属性。您的编程环境对一些基本的属性赋缺省值。下面是 Graph 控件的 Visual Basic 缺省属性页。



第二种属性页叫自定义属性页。随着控件的不同，自定义属性的外观和功能而不同。下面是 Graph 控件的自定义属性页。



### 1.3.2 编程设置

例如，在Visual Basic中对Boolean的值改变：

```
Boolean1.Value=0
```

建议大部分属性不提倡用编程设置，除非您对各属性非常透彻地了解。因为通过属性页设置属性非常地简单。

### 1.4 使用控件的方法

ActiveX控件有它自己的方法或函数，您可在您的程序中直接调用这些方法。：如在VB中：

```
Graph1.PlotGraphs value
```

### 1.5 使用控件的事件

当您在Form上配置完控件后，您可以编写它的事件例程。如：

在Visual Basic中：

```
Private Sub Graph1_Click()
```

```
End Sub
```

## 第二章 在 Visual Basic 中使用 Evoc ActiveDAS 控件

这一章描述怎样在 Visual Basic 中使用 Evoc ActiveDAS 控件, 怎样插入控件在 Visual Basic 环境中, 怎样设置它们的属性、方法和事件, 怎样操纵这些控件。

板卡控件使用前, 先要用本软件包中的配置工具 DevInst.exe 对卡的资源进行配置 (包括端口、中断、DMA)。使用控件前要先要注册 MFC 库, 如: regsvr32

C:\windows\system\mfc42.dll

### 2.1 开发 Visual Basic 应用程序

#### 2.1.1 把 Evoc ActiveDAS 控件装入工具箱

在建立应用程序之前, 必须把 Evoc ActiveDAS 控件加入到工具箱。所有的 Evoc ActiveDAS 控件被分为三类: EvocUI10.ocx, EvocEquipment.ocx, EvocChart10.ocx。加入哪一控件包根据您的应用程序的需要。

使用下列方法加入 Evoc ActiveDAS 控件到工具箱。

- 1) 在一个 Visual Basic 工程中, 在工具箱的空白处单击右键, 选择部件(components...)
- 2) 在弹出的对话框中。找到 EvocChart10 ActiveX Control module, EvocEquipment ActiveX Control module 和 EvocUI10 ActiveX Control module。
- 3) 在您所要的控件包的前面打上标记, 然后按 OK 键即可。

### 2.1.2 用 Evoc ActiveDAS 控件建立用户界面

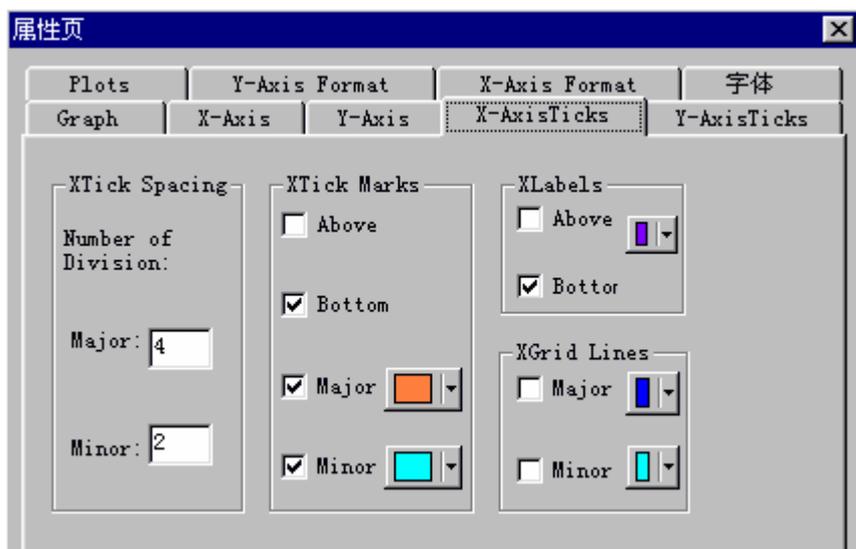
把Evoc ActiveDAS控件加入到工具箱后，就可以用它们建立您的应用程序了。选择相应的控件图标，从工具箱拖动到Form上，移动到合适的位置，调整好大小。然后通过属性页设置属性。

### 2.1.3 在 Visual Basic 中使用属性页

把控件放到Form上之后，Visual Basic会自动设置一些缺省值。可通过两种方式设置控件的属性。一是VB环境自带的，如下：



另一种是自定义式的，如下：



#### 2.1.4 用 Visual Basic 程序编辑属性

您可以通过程序对控件的属性进行设置和读出。例如：`Boolean1.value=0`；建议大部分属性不提倡用编程设置，除非您对各个属性非常透彻地了解。

#### 2.1.5 Visual Basic 中使用控件的方法

ActiveX控件有它自己的方法或函数，您可以在您的程序中直接调用这些方法。如：

```
Graph1.PlotGraphs value
```

#### 2.1.6 Visual Basic 中使用控件的事件

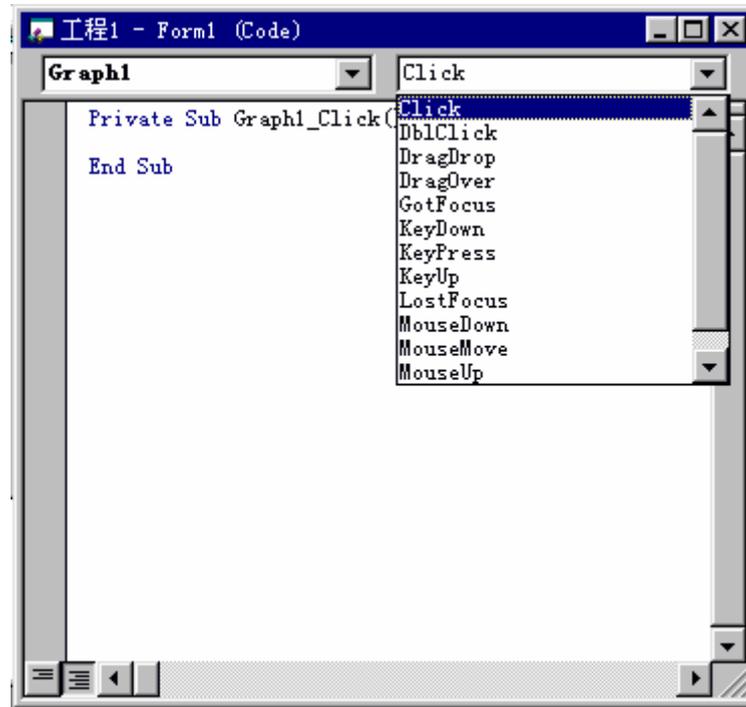
当您在Form上配置完控件后，您可以编写它的事件例程。如：

在Visual Basic中：

```
Private Sub Graph1_Click()
```

```
End Sub
```

如下图，右边可选择控件的事件，左边可改变控件。





### 第三章 在 Visual C++中使用 Evoc ActiveDAS 控件

这一章描述怎样在Visual C++中使用Evoc ActiveDAS控件，解释怎样插入Evoc ActiveDAS控件到Visual C++环境中去，怎样建立需要的包装类，怎样利用MFC应用向导建立应用程序，怎样用类向导建立控件的应用程序，讨论怎样操作控件。在这一点上，您应当对C++编程、Visual C++环境和ActiveX控件相当熟悉。

板卡控件使用前，先要用本软件包中的配置工具DevInst.exe对卡的资源进行配置（包括端口、中断、DMA）。使用控件前要先要注册MFC库，如：regsvr32

C:\windows\system\mfc42.dll

#### 3.1 开发 Visual C++应用程序

下面的过程怎样开始一个使用Evoc ActiveDAS控件的VC程序的开发。

- 1) 在Visual C++中建立一个新的工作空间或工程。
- 2) 为了建立使用Evoc ActiveDAS控件的程序，可使用MFC应用向导。
- 3) 当您建立好应用程序的框架之后，您需要把Evoc ActiveDAS控件加入到应用程序的工具条上。
- 4) 把控件加入到应用程序后，用属性页设置控件的属性。
- 5) 使用控件的方法对控件进行操作，使用控件的事件响应控件的动作。

##### 3.1.1 在 Visual C++中建立应用程序

建立应用程序时，在ActiveX Control的选择框上打上标记。

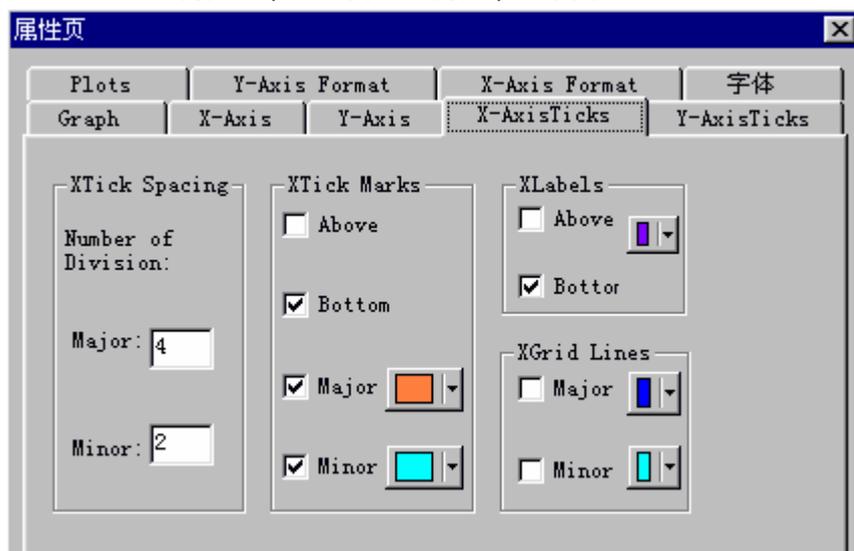
### 3.1.2 把 Evoc ActiveDAS 控件装入工具箱

当您把控件加入到部件库时，将在工程中生成一系列的C++包装类，通过这些包装类可实现对控件的各种操作。只有激活Visual C++的对话框编辑器时，控件工具条才出现。下面叙述加入控件到工程中的步骤。

- 1)选择Project>>Add To Project...>>Components and Controls，双击注册的ActiveX控件。
- 2)找到您所需要的控件，点Insert按钮，OK。

### 3.1.3 用 Evoc ActiveDAS 控件建立用户界面

在您把控件加入到工具条后，把所需的控件拖到对话框上，可以改变它的大小。点击右键，打开自定义属性页，设置自定义属性，如下图：



### 3.1.4 用 Evoc ActiveDAS 控件编程

可使用Visual C++的包装类对控件的属性、方法和事件进行编程。赋给控件一个变量，以后通过此变量对此控件操作。对于Graph和Chart控件，要用到Variant类型，比VB中的编程麻烦。具体可参考例子VCGraphDemo。

1) 先定义类型VARIANT如data，类型SAFEARRAY FAR\* 如psa。

2) 程序初始化时调用VariantInit(&data);

3) 程序销毁时调用

```
if(psa != NULL)
    SafeArrayDestroy(psa);
VariantClear(&data);
```

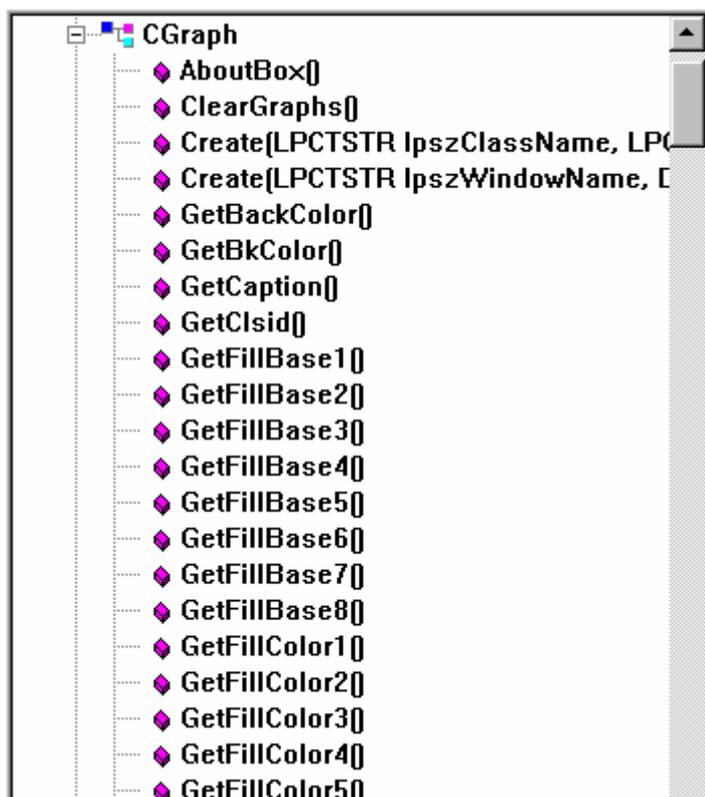
4) 在例子中的OnButton1()中的代码:先定义一个数组，如double array[1024];

再定义一个安全数组的维数描述数组，如SAFEARRAYBOUND rgsabound[1];表明安全数组为一维。

SafeArrayCreate 用来建立安全数组，  
SafeArrayPutElement对安全数组赋值

### 3.1.5 Visual C++中作用控件的属性

现在，您不能象VB那样直接对控件的属性进行读写，而须通过包装类的函数间接对控件的属性操作。例如：`m_graph1.GetCaption()`



3.1.6 Visual C++中作用控件的方法

例如，m\_graph1.PlotGraphs value

### 3.1.7 Visual C++中作用控件的事件

用下列步骤加入事件例程。

- 1) 右键单击控件，选建立类向导。
- 2) 找到Message Maps页面。
- 3) 选择Object ID。
- 4) 对其选择事件。
- 5) 加入事件对其编程即可。



## 第四章 在 Delphi 中使用 Evoc ActiveDAS 控件

这一章描述怎样在Delphi中使用Evoc ActiveDAS控件，您应当对Delphi环境有相当的了解。

板卡控件使用前，先要用本软件包中的配置工具DevInst.exe对卡的资源进行配置（包括端口、中断、DMA）。使用控件前要先注册MFC库，如：`regsvr32 C:\windows\system\mfc42.dll`

### 4.1 开发 Delphi 应用程序

#### 4.1.1 把 Evoc ActiveDAS 控件装入组件面板

在您建立应用程序之前，必须先把控件加入到Delphi环境中的部件面板中。您只需加入一次即可，不需要时可以显示地移走它们。而不象在VB和VC那样，每建立一个应用程序，就必需加入一次。当您加入控件到面板的时候，您就建立了Pascal的import units(相当于头文件)，它声明了控件的属性、方法和事件。您把控件加入到窗体时，对import units的参考就自动地加入到了程序。用下列步骤把控件加入到面板中：

- 1)在Delphi环境中，选Componet菜单，选Import ActiveX Control.....，这时将弹出所有注册的控件的窗口。
- 2)选择所需要的控件组，点Install.....
- 3)这时Delphi环境在\Import 路径下产生它的头文件。
- 4)在安装对话框中，点击OK即可。
- 5)在下面的对话框中，点YES即可，Delphi会自动编

绎刚加入的控件包文件。若控件没注册，必须先注册。

## 第五章 Evoc ActiveDAS 控件的详细介绍

板卡控件使用前,先要用本软件包中的配置工具DevInst.exe对卡的资源进行配置(包括端口、中断、DMA)。使用控件前要先要注册MFC库,如:regsvr32 C:\windows\system\mfc42.dll。

### 5.1 E7Segment 控件



七段数码管用来模仿 LED 显示数字的器件。



Value属性设置其要显示的数值, DigitNum为数值的位数, DigitNum After Point为小数点后的位数, PreZero是否有前置零, Signed是否显示符号。



declined是否倾斜，show unused segment时是否用阴影显示未使用的段。Border可改变控件是否有边框。同时可改变控件的背景色和前景色。当想改变此控件的显示数值时，在程序直接改变Value属性值即可，控件会自动刷新其显示。

## 5.2 Boolean 控件



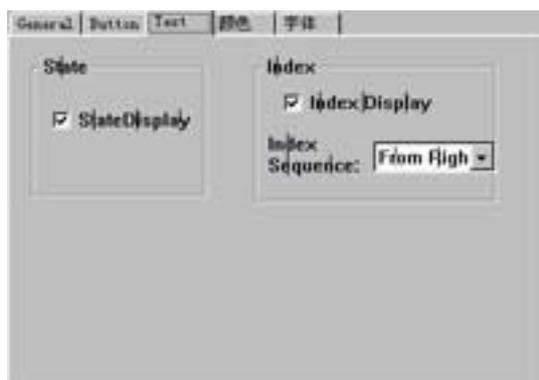
布尔型控件是显示和操作开关量的控件，对于数字量的输入输出非常地便捷。Caption为控件的标题。



它最多可达50位，由Button Num属性设定。属性Distance表示按钮之间的距离。Direction表示按钮的排列方向。Only One On Flag表示所有按钮中，只有一个为开，其余为关闭状态。



Index Sequence表示索引的方向，Index Display表示是否显示索引，State Display表示是否显示按钮的开关状态。



当您用鼠标点击控件的按钮时，其Value值将作相应的变化，可加入Click事件，以实现控制的功能。当然，也可实现采集数据的功能。具体编码可参考DemoUI10DIO，此例子是基于本公司的多功能卡PLG-812PG的数据采集和控制的应用。

### 5.3 Knob 控件



旋钮控件以表盘样式，可输出和采集数据，包括模拟量或数字量。

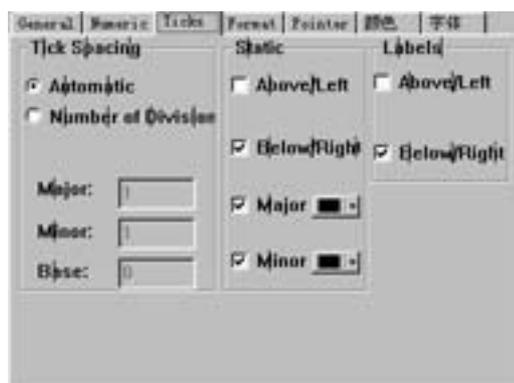


上面的属性页，Caption为控件的标题，Border表示控件是否有边框，DigitDisplay在控件底部是否显示当前控件的值。

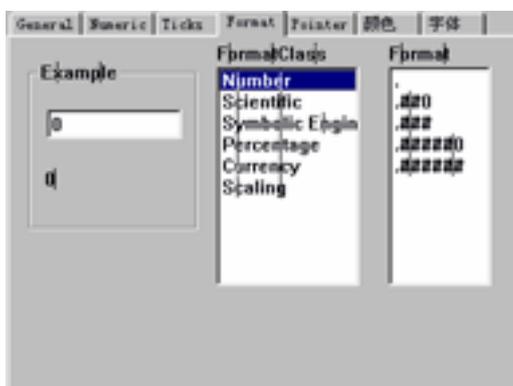
下面的属性页中，Minimum 为刻度的最小值，Maximum 为刻度的最大值，Inversse表示最大值与最小值是否互换。



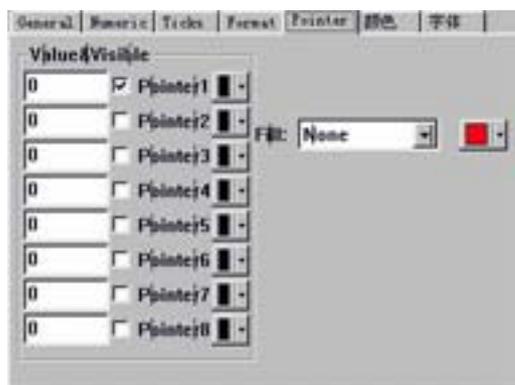
刻度间隙有两种类型：Automatic和Number of Division。Automatic 系统自动为你分配好刻度，而Number of Division需要自定义。Major为全部主刻度格数，Minor为每一主刻度格之内次刻度格数。Static框内表明了刻度线的位置和颜色。而Labels框内表明了刻度标签的位置。



Format属性页定义了许多种数据的格式。刻度标签和当前值的显示将以您选定的格式显示。Example用来检验此种格式的即时效果。共有6大类格式，每一类又有数种格式。这6大类为数值型(Number)、科学型(Scientific)、工程符号(Symbolic Engineer)、百分比型(Percentage)、货币型(Currency)、规模型(Scaling)。具体每一大类的更详细的格式在此不作详细介绍。



它最多可达8个指针（8个通道）。可分别对每个指针的颜色和值进行设置。注意，不倡导对控件的Value属性设置，只建议分别对每个指针的Value设置，整个控件的Value属性将等于刚刚设置的那个指针的Value值。



Knob 控件的使用可参考例子 DemoUI10DA 和 DemoUI10AD。一个用于数据显示，一个用于数据采集。Format 属性页用于数据的显示格式，用户可根据不同的需要进不同的设置。Numeric 设置刻度盘的范围。在 Ticks 属性页中，有 Tick Spacing 选项。

#### 5.4 Slide 控件

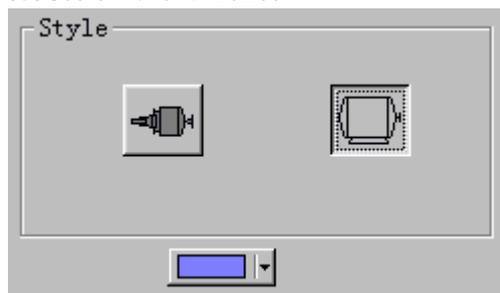


拉动条控件与旋钮控件的属性基本相同，除了表现形式不同之外。具体使用方法参见例子 DemoUI10DA 和 DemoUI10AD。

#### 5.5 Motor 控件



有两种形式的马达控件：

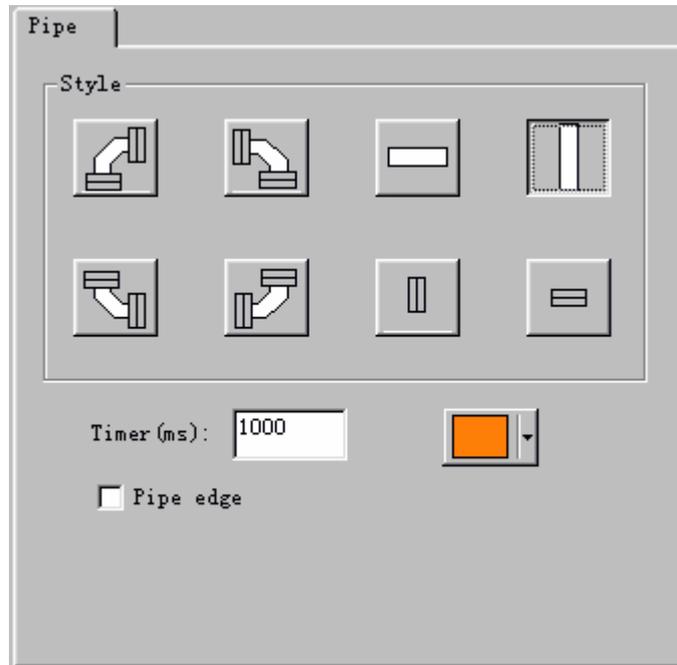


可设置其颜色。使用方法参见例子 synthesize。

#### 5.6 Pipe 控件



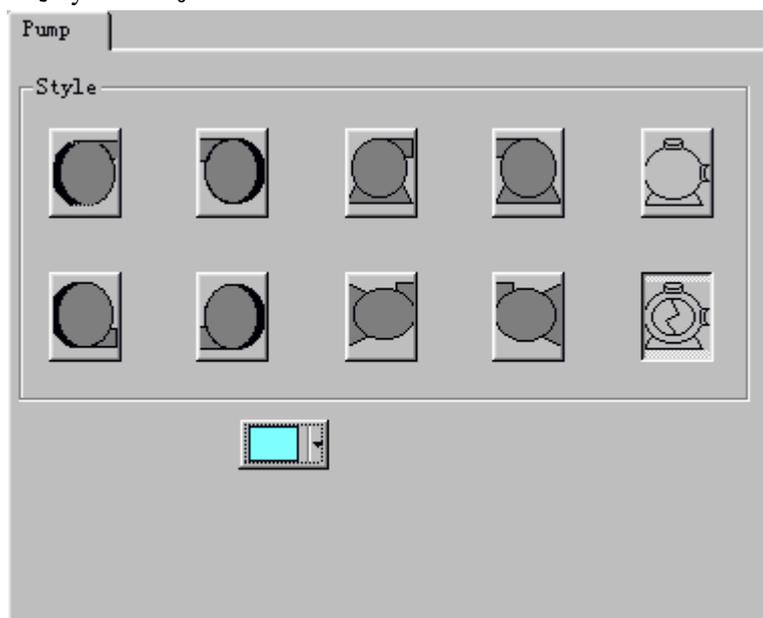
有8种形式的管子控件，Timer 可设置其颜色和闪烁的频率，间隔时间为毫秒。Pipe edge 为管子为直管子的中间段时两端是否有边界。使用方法参见例子 synthesize。



## 5.7 Pump 控件



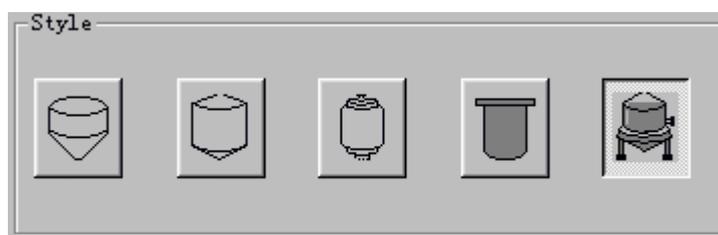
泵控件有10种样式，可设置其颜色，具本使用可参考例子synthesize。



## 5.8 Tank 控件



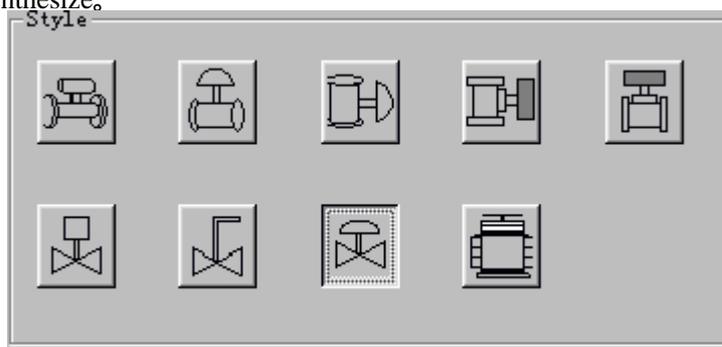
罐控件有5种样式，可设置其颜色。具本使用可参考例子synthesize。



## 5.9 Value 控件



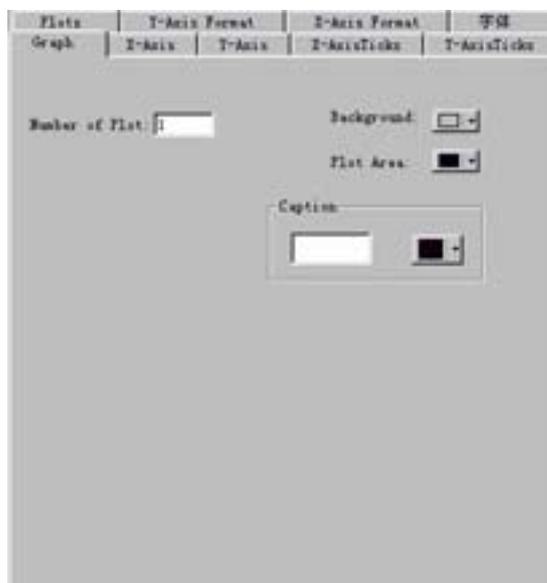
阀控件有9种样式，可设置其颜色。具本使用可参考例子  
synthesize。



## 5.10 Graph 控件



Graph控件是输出历史数据的复杂控件。Number of Plot  
表示几个输出。Caption为控件的标题，Background为控  
件的背景色，Plot Area为输出区域的颜色。



View Number属性为控件中X轴上显示的数据个数，Start Number为从数组中的第几个数据开始显示。

Flats	Y-Axis Format	Y-Axis Format	字体
Graph	Y-Axis	Y-Axis	Y-AxisTicks

View Number:

Start Number:

Minimum与Maximum为Y轴上刻度的最小值与最大值。

Flats	Y-Axis Format	Y-Axis Format	字体
Graph	Y-Axis	Y-Axis	Y-AxisTicks

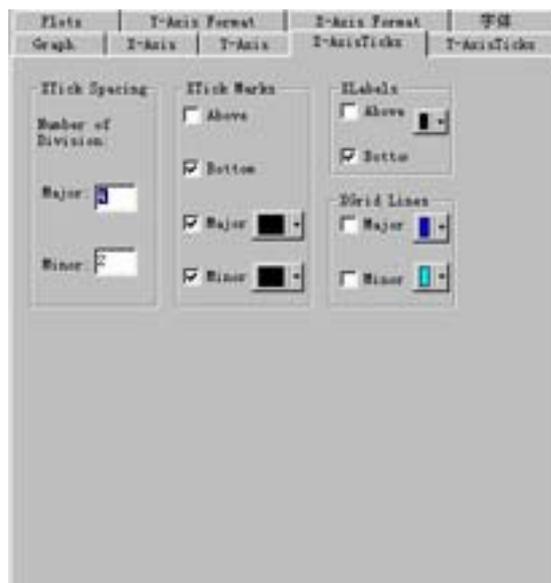
  

Scale

Minimum:

Maximum:

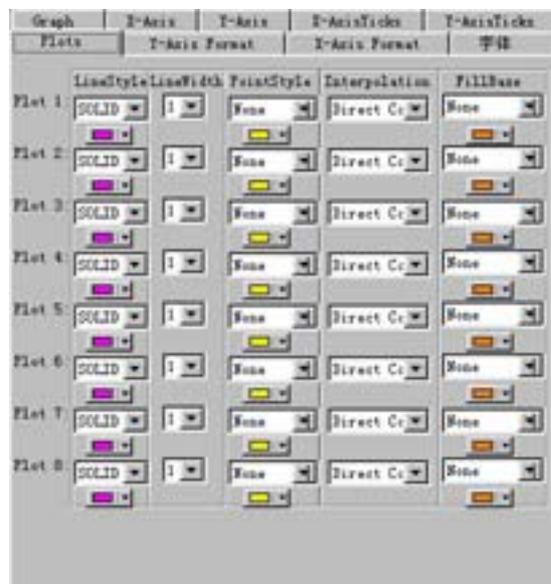
在X-AxisTicks属性页刻画了X轴上的刻度的特征。XTick Spacing表示主次刻度的格数,其中, Major为所有的主刻度的格数,Minor为每一主刻度格子所包含的次刻度的格数。XGrid Lines表示在X轴上是否画格子线。Y-Axis Ticks属性页表示Y轴上的属性,可参考X-Axis Ticks属性进行设置。而Xtick Marks表明了刻度线的位置和颜色。Xlabels表示刻度标签的位置。Xgrid Lines表示了X轴方向是否打上网格线,及其颜色是什么。Y-AxisTicks刻画了Y轴上的刻度的特征。具体设置可参照X-AxisTicks属性页设置。



Y-Axis Format属性页与Knob控件的Format属性页基本相同，可参考其设置。X-Axis Format属性页表示X轴上的刻度标签格式。共有三大类：数值型(Number)、时间型(Time)和日期型(Date)。数值型分四种，有十进制、十六进制、八进制和二进制。时间型和日期型又分别有两种。其中，Base为对应数所数组中的第一个数据，为基址时间。Interval为间隔的数据个数，表现在格式上的单位为秒或天。



它最多可有8个输出。这些所有要输出的数须存放在同一数组中。Plots属性定义了各个输出的样式，包括线型(LineStyle)、线宽(LineWidth)、点的形状(PointStyle)、点间的连接方式(Interpolation)、填充方式(FillBase)、颜色等。



属性设置好之后，可按照例子GraphDemo编程。参考下列步骤：

- 1)建立double型全局变量数组Gdata，数组长度根据需要而定。
- 2)对数组进行赋值。
- 3)输出数据，Graph1.PlotGraphs GData  
也可加上个ScrollBar拖动查看输出。也可设置多个输

出。用ClearGraphs可清除输出。

### 5.11 Chart 控件



Chart控件是画实时曲线的控件，它最多可有8个输出。Chart控件的属性页与Graph控件的属性页基本一样，可参照其进行属性设置。其中，在Chart属性中，Plot的类型有Overlaid(覆盖式)和Stacked(堆栈式)。更新方式有3种：Strip(条状)、Scope(帧状)、Sweep(扫描状)。History为存储历史数据的缓冲区的大小，调用InitControl初始化控件时将根据此大小开辟内存。Caption为控件的标题，Background为控件的背景色，Plot Area为输出区域的颜色，Number of Plot为输出几条线。



属性设置好之后，可按照例子ChartDemo编程。参考下列步骤：

- 1)建立double型全局变量数组Value，长度为要输出的个数。即使长度为1，也须定义成数组的形式。
- 2)用InitControl方法初始化控件，并初始化数组。
- 3)在Timer事件中先输出数据（为控件开辟内存），如  
Chart1.PlotCharts Value
- 4)在Timer事件中对数组重新赋值，注意3)和4)不要颠倒。
- 5)程序结束时一定要用ReleaseControl释放内存。

## 5.12 DASDevice 控件



DASDevice控件可以产生一个选择设备的对话框，并获此设备的设备号。它有两个主要属性：DeviceName和DeviceNumber。其中，DeviceName为只读属性。DeviceNumber为选择后的设备号。这些设备的使用之前应先用工具软件DevInst.exe配置各个设备的资源状况，并注意不要使它们的资源冲突。其中ISA-HG16AD卡的资源配置要在控制面板中的设备管理器中进行添加新硬件，并配置。此控件还有一重要的方法SelectDevice用于产生选择设备的对话框。

## 5.13 DASAI 控件



用于模拟量的输入。



上面为其属性页。可通过Device按钮选择设备，也可通过DASDevice控件选择设备。两者选一种方式即可。

EventEnable表示采集完数据后是否要触发事件。TransferMode最多有三种传输模式：软件轮询、中断、DMA，不同的设备可能有所不同。同样，触发方式也有三种：软件、外部、时钟定时触发。OverallInputRange或AdGain用来设置卡的输入范围，具体使用要参看这块卡的使用说明书。StartChannel表示要采集哪个通道。InputMode属性有两种，单端输入或双端输入。当进行批量数据采集时，要设置采样速率（SampleRate）和采样个数（NumberOfSamples）正确设置好这些属性后，就可通过控件操作硬件了。

它有两个事件，当属性EventEnable为真时，数据采集完后OnEventRaw事件产生，可通过参数Data获得原始数据，参数DataCount为采集的个数。采集完成后或停止采集时将触发OnTerminated事件，可在此事件中处理采集善后工作。

它有四个主要的方法。各属性设置好后，先调用方法OpenDevice打开设备。OpenDevice成功返回真，否则为假。其它控件类似。RawInput为原始数据输入的函数。调用一次只采集一次数据。要想批量采集某通道的数据，须调用AcquireStart和AcquireStop方法，在事件OnEventRaw中取得批量数据。

可参考例程DASAI Demo使用。

使用步骤如下：

- 1) 加入Evoc DASAI和Evoc DASDevice控件。
- 2) 对DASAI控件点右键,选特性...,弹出属性页对话框。
- 3) 设置各属性，可通过Device按钮选设备，也可通过

DASDevice控件选设备。

- 4) 打开DASAI。
- 5) 一次性采集只调用RawInput方法即可。
- 6) 批量采集AcquireStart方法。

此控件所持有的卡的种类请参考第七章。

控件使用前，先要用本软件包中的配置工具 DevInst.exe 对卡的资源进行配置（包括端口、中断、DMA）。

#### 5.14 DASAO 控件



模拟量输出的控件。

控件使用前，先要用本软件包中的配置工具 DevInst.exe 对卡的资源进行配置（包括端口、中断、DMA）。

属性页如下：



选择设备可通过属性页上的按钮或通过DASDevice控件。例程通过DASDevice控件选择设备。

Channel属性为选择模拟量输出的某个通道，可通过属性页选择。

方法RawOutput为原始数据输出。把模拟量对应的值0~44095作为函数的参数进行输出。输出成功返回真，否则为假。

可参考例程DASADemo编程。

此控件所持的卡的种类请参考第七章。

### 5.15 DASDI 控件



用来数字量的输入。

控件使用前，先要用本软件包中的配置工具 DevInst.exe

对卡的资源进行配置（包括端口、中断、DMA）。

属性页如下：



它有两个重要的属性Port( 端口 )和Bit( 端口的第几位 )。例如，720有32通道（即32位），则端口数为4（0~3），每个端口有8位， $4 \times 8 = 32$ 。

设置好属性后，程序开始要调用方法OpenDevice。它有两个重要方法BitInput和ByteInput。BitInput要设置好属性Port和Bit，其对应卡的某个通道。如Port为0，Bit为4，则对应卡的第四通道。当此通道为1时，此方法返回值为真。否则为假。而方法ByteInput就无需设置属性Bit，仅

设置属性Port即可。一个通道对应一个字节，即读入8位（8个通道）。

可参考例程DASDIDemo编程。

此控件所持的卡的种类请参考第七章。

#### 5.16 DASDO 控件



此控件为数字量输出控件。

控件使用前，先要用本软件包中的配置工具 DevInst.exe 对卡的资源进行配置（包括端口、中断、DMA）。

属性页及属性的设置完全同控件 DASDI。

ByteOutput 为字节数字量输出，这时要设置好属性 Port。输出成功返回真，否则返回假。

BitOutput 为位数字量输出，要首先设置好属性 Port 和 Bit。参数为布尔型，当输出 1 时，使参为真值；反之则为假值。此方法输出成功时返回真值，否则返回假值。

此控件还有不同于 DASDI 控件的两个方法 BitReadBack 和 ByteReadBack，使用方法与 BitOutput 和 ByteOutput 差不多。注意，有些卡可能不支持回读，具体请参考此卡的使用手册。因此，这两个回读方法并不适用于所有数字量输出的卡。

可参考例程DASDODemo编程。

此控件所支持的卡的种类请参考第七章。

#### 5.17 DASCounter 控件



控件使用前，先要用本软件包中的配置工具 DevInst.exe 对卡的资源进行配置（包括端口、中断、DMA）。

用于支持计数器功能的ISA卡。它有2个得要的属性 Channel各PresetValue。Channel为某计数器的通道值。一般一片8254有三个计数器，即有三个通道。PresetValue 为某计数器开始时设定的预置值。这2个属性可在属性页

中预设好。方法EnableCounter用来启动或停止计数器。参数为True时启动。参数为False时停止。  
当读取CounterValue(只读属性)的值时，控件将返回相应计数器的值。

#### 5.18 DASPulse 控件



控件使用前，先要用本软件包中的配置工具 DevInst.exe 对卡的资源进行配置（包括端口、中断、DMA）。  
用于计数器的脉冲输出功能。Channel的意义同控件DASCounter。而PulseWidth提供给计数器时钟信号周期的个数。如时钟信号为10KHz，则周期为0.1毫秒。若PulseWidth为4，则输出脉冲信号的周期为0.4毫秒。当然，输出频率相应为2.5KHz。具体参考8254的有关资料。  
Channel和PulseWidth属性可通过属性页设置。设置好后，可调用方法EnablePulseOut来启动或停止输出。

#### 5.19 DASAlarm 控件



报警控件。方法ResetAlarm用于复位报警控件。属性AlarmMode有两个选项（momentary和Latched）。当为Momentary时，超过上限（HiLimit）或下限（LoLimit）时，控件处于报警状态。只要这时EnableAlarm为True，就可激必OnHiAlarm，OnHiToNormal，OnLoAlarm，OnLoToNormal中的某些事件，当Value值回到正常状态时，控件将自动解除报警状态。再次超限，再次报警。  
当AlarmMode为Latched方式时，一旦超限处于报警状态，即使Value值回到正常值，仍然锁定于报警状态，除非设置RetriggerAlarm为True时，才可关闭此次报警。

Value为向DASAlarm提供的实时数据值。

OnHiAlarm为由正常状态越过HiLimit时而引发的事件。

OnHiToNormal为由超限报警回到正常时而引发的事件。

OnLoAlarm为由正常状态越过LoLimit时而引发的事件。

OnLoToNormal为由超限报警回到正常时而引发的事件。

#### 5.20 PCL731A 控件



此控件用于数字量的输入输出，专用于PCL731(A)卡。

它有以下方法：

Init为初始化函数。

InData，读取某通道的值。

OutData，输出数字量。

ConfigCHN，对某通道配置。它有两个参数。Channel表示哪个通道，config\_value为寄存器配置值。

SetINTOp为设置中断的操作模式。Directory表示方向，输入或输出。Buffer为存储数字的缓冲区。

INTStart，中断操作开始。INTStop停止中断操作。

GetINTStatus取得中断的状态，interrupt\_count为即时的传输数据的个数。

#### 5.21 PCL733 控件



此控件用于数字量的输入，专用于PCL733卡。它有以下方法：

Init为初始化函数。

InData，读取某通道的值。

SetINTOp为设置中断的操作模式。Directory表示方向，输入或输出。Buffer为存储数字的缓冲区。

INTStart，中断操作开始。INTStop停止中断操作。

GetINTStatus取得中断的状态，interrupt\_count为即时的传输数据的个数。

#### 5.22 PCL734 控件



此控件用于数字量的输出，专用于PCL734卡。它有以下方法：

Init为初始化函数。

OutData, 输出数字量。

### 5.23 PCL836A 控件



计数器或定时器, 专用于PCL836(A)卡。

DeviceOpen, 打开设备, 支持多个设备。有设备号和基地址两个参数。在调用其它方法之前必须选调用此方法。

DeviceClose, 程序结束时须关闭设备。

CounterConfig, 参数Counter为计数器号, 0~5。参数MaxInFreq为计数器的最大输入频率。当为0时计数器无效。其最大值为312kHz。

CounterEventStart计数器对外部事件开始计数。Count为计数器的值。

CounterEventRead读取外部事测量值。参数Counter为计数器号, 0~5。Overflow标志是否溢出。Count为计数器的值。

CounterFreqStart频率计数开始。

CounterFreqRead, 参数Counter为计数器号, 0~5。

FreqLevel频率级别, =0 MAX 1Hz~65kHz, =1 MAX 10kHz~650kHz, =2 MAX 100kHz~6500kHz。Freq读取的频率值。

CounterPulseStart脉冲输出开始。参数Period为周期(毫秒), 参数UpCycle为高电平长度(毫秒)。

FreqOutStart频率输出开始, 参数Counter为计数器号, 0~5。Fout为要输出的频率值。

FreqOutReset频率输出复位。

CounterReset计数器复位。

DioWritePortByte数字量写端口字节。参数Mask为屏蔽字, 参数State为要写的值。

DioWriteBit数字量写位输出。参数Bit为哪一位，参数State为要输出的值，只能为0或1。

DioGetCurrentDOByte回读当前输出的字节。参数Port标识要回读的端口，参数Value为回读的值。

DioGetCurrentDOBit回读当前输出的某位的值。参数Bit为哪一位，参数State为要回读的值，只能为0或1。

DioReadPortByte数字量读端口字节。参数Mask为屏蔽字，参数State为要读的值。

DioReadBit数字量写位输入。参数Bit为哪一位，参数State为要读取的值，只能为0或1。

## 第六章 例子及注意事项

### 6.1 DemoUI10DIO 例子

本例是基于本公司生产的多功能卡PLC-812PG，实现数字量的输入输出功能。需首先安装它的驱动程序(DLL2)才能运行此例。上面的Boolean控件为输出，共有8个通道。起始值为00000000（二进制）。下面的Boolean控件为数据输入，共有8个通道。

1)先把PLC-812PG驱动程序的头文件Dll2.bas

2)定义一个全局变量 Dim PCL\_BaseAddress As Long

3)先初始化812PG

```
Private Sub Form_Load()  
    '初始化812PG  
    PCL_BaseAddress = &H220  
    If W_812PG_Initial(CARD_1, PCL_BaseAddress) <>  
ERR_NoError Then  
        MsgBox "PCL-812PG Initialization Error!"  
        Unload Form1  
    End If  
End Sub
```

4)采集的编程非常简单，在Timer事件中。

```
Private Sub Timer1_Timer()  
    Dim di_data As Byte  
    W_812PG_DI di_low_byte, di_data '采集数字量  
    If Boolean2.Value <> di_data Then  
        Boolean2.Value = di_data '让控件显示数字量
```

```
End If
```

```
End Sub
```

5)在Boolean1的Click事件中有输出数字量的代码

```
Private Sub Boolean1_Click()
```

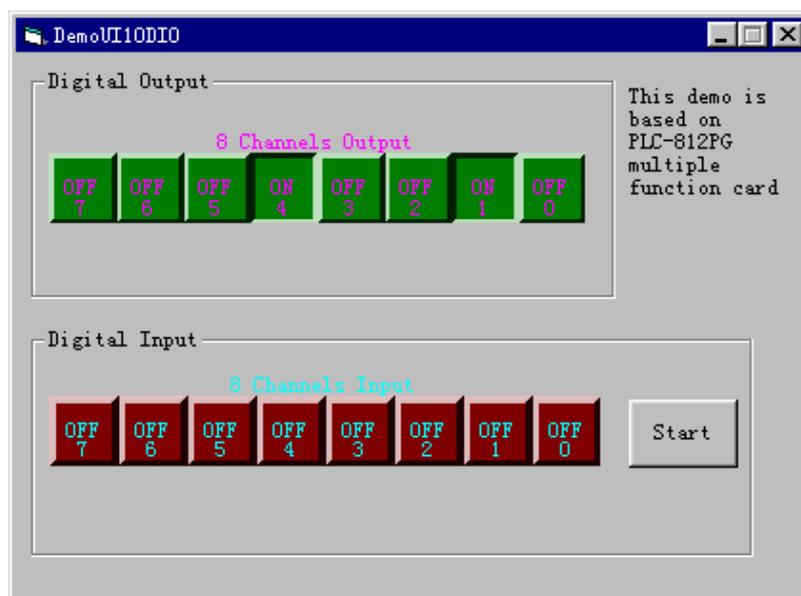
```
Dim do_data As Byte
```

```
do_data = Boolean1.Value
```

```
W_812PG_DO DO_Low_BYTE, do_data '根据控件的值输出  
数字量
```

```
End Sub
```

本例主要描述了Boolean控件的使用方法。用户编程时可参照此例进行编程。下图是本例中的一个界面。



## 6.2 DemoUI10DA 例子

本例是基于本公司生产的多功能卡PLC-812PG，实现模拟量的输出功能,即实现数模转换。需首先安装它的驱动程序(DLL2)才能运行此例。共有2个通道。只要用鼠标拖动Knob控件或Slide控件上的指针，对应通道就会相应的模拟量的输出。对某指针的拖动，写在事件MouseMove中。

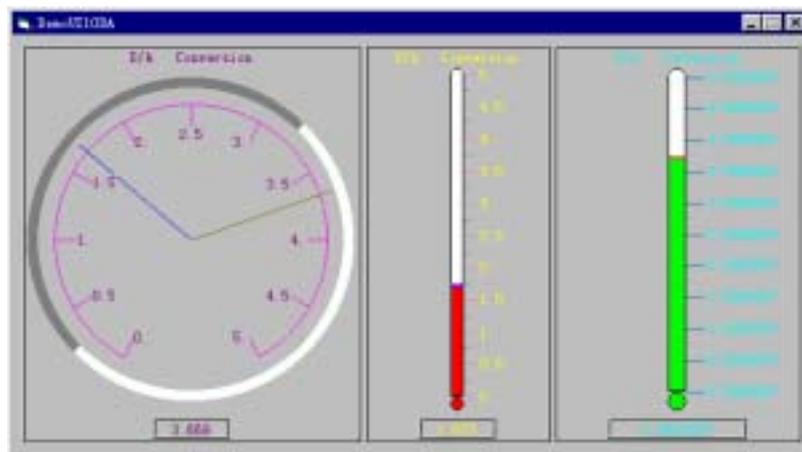
- 1) 先把PLC-812PG驱动程序的头文件Dll2.bas
- 2) 定义一个全局变量     Dim PCL\_BaseAddress As Long
- 3) 初始化812PG
 

```
Private Sub Form_Load()
'初始化812PG
PCL_BaseAddress = &H220
If W_812PG_Initial(CARD_1, PCL_BaseAddress) <>
ERR_NoError Then
MsgBox "PCL-812PG Initialization Error!"
Unload Form1
End If
End Sub
```
- 4) 通过移动Knob中的指针来进行数模转换
 

```
Private Sub Knob1_MouseMove(Button As Integer, Shift As
Integer, x As Single, y As Single)
If volt1 <> Knob1.Pointer1Value Then
volt1 = Knob1.Pointer1Value
Slide1.Pointer1Value = Knob1.Pointer1Value
da_data1 = volt1 / 5 * 4095 'D/A Range == -5V to +5V
W_812PG_DA DA_CH_1, da_data1 '在812PG的通道1
输出模拟量
```

```
End If
If volt2 <> Knob1.Pointer2Value Then
    volt2 = Knob1.Pointer2Value
    Slide2.Pointer1Value = Knob1.Pointer2Value
    da_data2 = volt2 / 5 * 4095 ' D/A Range == -5V to +5V
    W_812PG_DA DA_CH_2, da_data2 '在812PG的通道2输出模拟量
End If
End Sub
5) 在Slide控件中移动指针进行数模转换
Private Sub Slide1_MouseMove(Button As Integer, Shift As
    Integer, x As Single, y As Single)
    If volt1 <> Slide1.Pointer1Value Then
        volt1 = Slide1.Pointer1Value
        Knob1.Pointer1Value = Slide1.Pointer1Value
        da_data1 = volt1 / 5 * 4095 ' D/A Range == -5V to +5V
        W_812PG_DA DA_CH_1, da_data1 '在812PG的通道1输出模拟量
    End If
End Sub
```

用户编程时可参照此例进行编程。下图是本例的中的一个界面。



### 6.3 DemoUI10AD 例子

本例是基于本公司生产的多功能卡PLC-812PG,实现模数转换采集数据的功能。需首先安装它的驱动程序(DLL2)才能运行此例。也是2个通道。

- 1) 先把PLC-812PG驱动程序的头文件Dll2.bas
- 2) 定义一个全局变量 Dim PCL\_BaseAddress As Long
- 3) 初始化812PG

```
Private Sub Form_Load()
```

```
    '初始化812PG
```

```
    PCL_BaseAddress = &H220
```

```
    If W_812PG_Initial(CARD_1, PCL_BaseAddress) <>
```

```
        ERR_NoError Then
```

---

```
MsgBox "PCL-812PG Initialization Error!"
```

```
Unload Form1
```

```
End If
```

```
W_812PG_AD_Set_Gain AD_GAIN_1
```

```
W_812PG_AD_Set_Mode AD_MODE_1
```

```
ad_count = 1
```

```
ad_start_ch = 0
```

```
ad_stop_ch = 1
```

```
End Sub
```

4) 采集数据的代码写在Timer事件中。

```
Private Sub Timer1_Timer()
```

```
Dim ch, ad_data As Long
```

```
Dim err As Integer
```

```
Dim volt As Single
```

```
Dim buffer(15, 99) As Integer
```

```
W_812PG_CLR_IRQ '清除中断
```

```
For ch = ad_start_ch To ad_stop_ch
```

```
W_812PG_AD_Set_Channel ch
```

```
err = W_812PG_AD_Aquire(ad_data)
```

```
If err <> ERR_NoError Then
```

```
MsgBox "PCL-812PG Error Number :" + Str(err)
```

```
Else
```

```
volt = ad_data
```

```
volt = 10 * volt / 4095 + (-5)
```

```
If ch = 0 Then 把采集得到的1通道的数据用控件显示出来
```

```
Knob1.Pointer1 Value = volt
```

```
Slide1.Pointer1Value = volt
E7Segment1.Value = volt
Else
  If ch = 1 Then    '把采集得到的2通道的数据用控件
                  显示出来
    Knob1.Pointer2Value = volt
    Slide1.Pointer2Value = volt
    E7Segment2.Value = volt
  End If
End If

End If
Next ch

End Sub
```

用户编程时可参照此例进行编程。下图是本例中的一个界面。



#### 6.4 GraphDemo 例子

本例模拟两条正弦曲线的输出。

1) 先在Form\_Load()中模拟生成正弦曲线的数据。

```
Private Sub Form_Load()  
    Dim k As Integer  
    deltaX = 3.14159256 * 4 / 210  
    For i = 0 To 4095 Step 2  
        k = i / 2
```

$GData(i) = (\sin(k * \text{deltaX}) * 10)$  '产生正弦曲线的数据，存放于数组中'

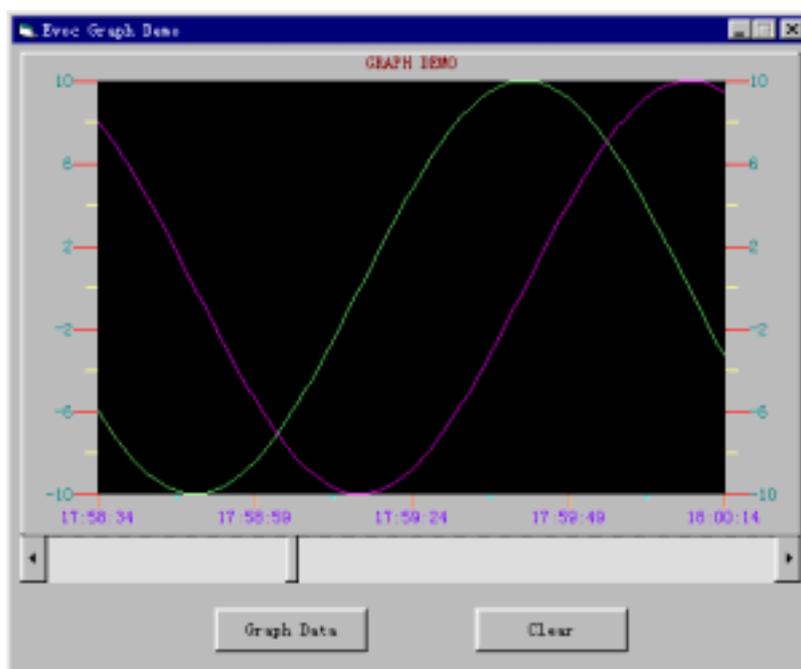
$GData(i + 1) = (\cos(k * \text{deltaX}) * 10)$  '产生余弦曲线的数据，存放于数组中'

Next i

End Sub

2) Graph1.PlotGraphs GData '根据数组中的数据输出曲线'

此例可拖动滚动条查看曲线，还可清除曲线。此控件的编程非常方便，用户可参照此例编程。下图是本例中的一个界面。



## 6.5 ChartDemo 例子

本例罗列了5个Chart控件，同样是模拟正弦曲线的方式显示了控件的不同样式。

- 1) 设置好各个属性之后，定义一个一维数组Value，长度为2。

```
Dim Value(0 To 1) As Double
```

- 2) 然后在Form\_Load()中调用

InitControl方法初始化控件（开辟内存），再对数组赋一个初值。

```
Private Sub Form_Load()
```

```
'初始化控件
```

```
Chart_Strip.InitControl
```

```
Chart_Scope.InitControl
```

```
Chart_Sleep.InitControl
```

```
Chart_Overlaid.InitControl
```

```
Chart_Stacked.InitControl
```

```
icount = 0
```

```
deltaX = 3.14159256 * 4 / 51
```

```
'初始化数组
```

```
Value(0) = 5 + (Sin(icount * deltaX) * 5)
```

```
Value(1) = 5 + (Cos(icount * deltaX) * 5)
```

```
End Sub
```

- 3) 千万不要忘记在Form\_Unload()中调用ReleaseControl方法释放控件（释放内存）。

```
Private Sub Form_Unload(Cancel As Integer)
```

```
'程序结束时一定要释放Chart控件的内存
```

```
Chart_Strip.ReleaseControl
```

```
Chart_Scope.ReleaseControl  
Chart_Sleep.ReleaseControl  
Chart_Overlaid.ReleaseControl  
Chart_Stacked.ReleaseControl
```

```
End Sub
```

- 4) 在Timer事件中先输出数组中的数据，再对数组重新赋值。注意顺序不要颠倒。

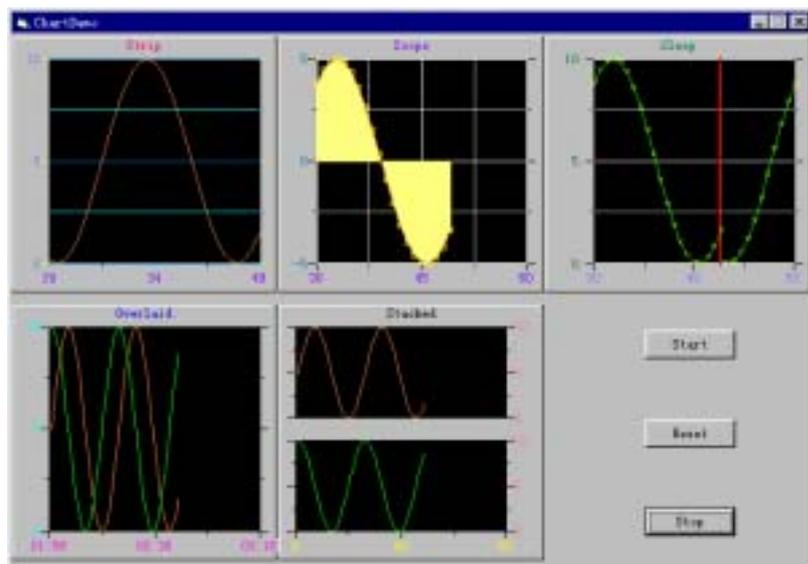
```
Private Sub Timer1_Timer()  
Dim Scope_Value(0) As Double  
'输出数据  
Chart_Strip.PlotCharts Value  
Scope_Value(0) = Value(0) - 5  
Chart_Scope.PlotCharts Scope_Value  
Chart_Sleep.PlotCharts Value  
Chart_Overlaid.PlotCharts Value  
Chart_Stacked.PlotCharts Value
```

Value(0) = 5 + (Sin(icount \* deltaX) \* 5) '产生正弦曲线的实时数据，存于数组中

Value(1) = 5 + (Cos(icount \* deltaX) \* 5) '产生余弦曲线的实时数据，存于数组中

```
icount = icount + 1  
If icount = 102 Then  
icount = 0  
End If  
End Sub
```

此控件的编程非常方便，用户可参照此例编程。下图是本例的中的一个界面。



### 6.6 Instrument 综合例子

本例模拟采集温度的。两个旋钮用来调节幅度的上限和下限。注意调节时，上限一定要大过下限。温度计实时显示当前的温度，曲线控件用来显示随时间变化的温度。

1) 需要定义一全局变量数组Value，大小为3。Value(0)为实测值，Value(1)和Value(2)为上下限。

Dim Value(2) As Double '定义一数组

2) 同样，不要忘了控件的初始化以及数组的初始化。

```
Private Sub Form_Load()
```

```
Chart1.InitControl '初始化Chart控件
```

```
Value(0) = 24
```

```
Value(1) = 24
```

```
Value(2) = 27
```

```
End Sub
```

3) 不要忘记释放。

```
Private Sub Form_Unload(Cancel As Integer)
```

```
Chart1.ReleaseControl '释放控件
```

```
End Sub
```

4) 在Timer事件中要先输出数据，再赋值。还要注意数据类型要匹配，需要转化数据类型时就要正确地转化。

```
Private Sub Timer1_Timer()
```

```
Chart1.PlotCharts Value
```

```
Slide1.Pointer1 Value = Value(0)
```

```
Randomize Second(Time) '初始化随机种子
```

```
Dim temp As Double
```

```
temp = ((35 * Rnd) + 20)
```

```
While (Abs(temp - (Value(2) + Value(1)) / 2) >  
Abs(Value(2) - Value(1)))
```

```
temp = ((35 * Rnd) + 20)
```

```
Wend
```

```
Value(0) = temp '随机产生数据，模拟温度值
```

```
If Value(0) > 35 Then
```

```
Value(0) = 35
```

```
Else
```

```
If Value(0) < 20 Then
```

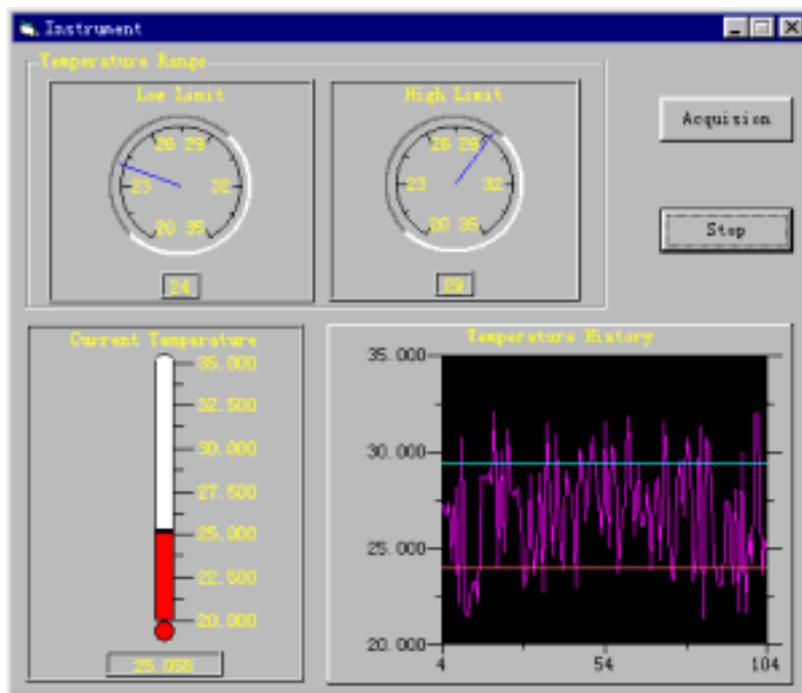
```
Value(0) = 20
```

```
End If
```

End If

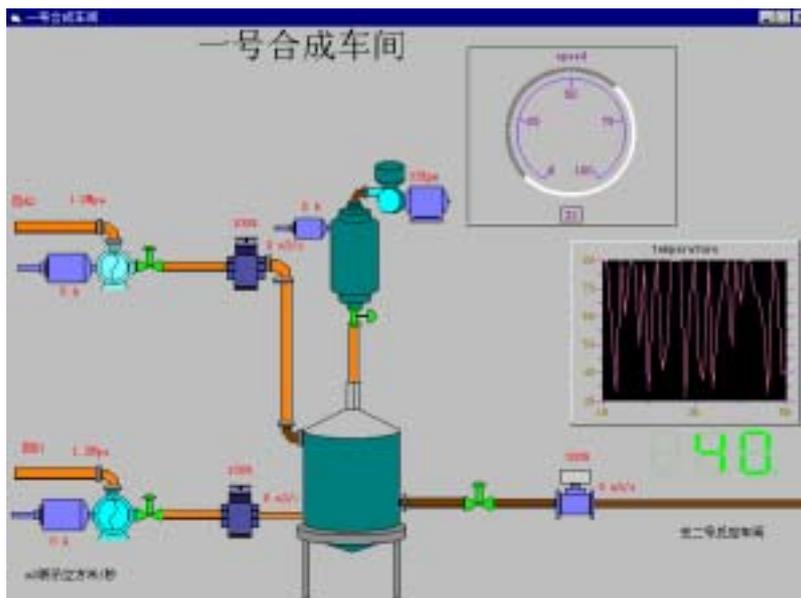
End Sub

此控件的编程非常方便，用户可参照此例编程。下图是本例中的一个界面。



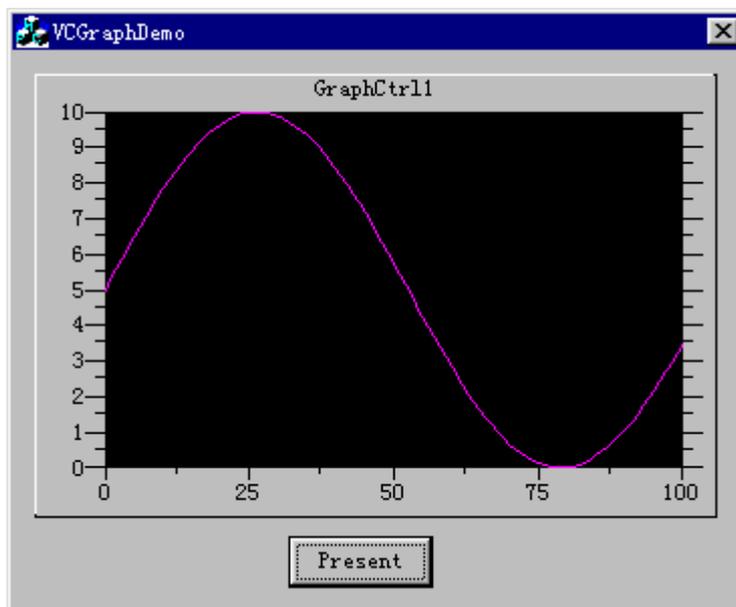
### 6.7 Synthesize 综合例子

本例子是一个化工合成车间的模拟图，主要用了 EvocEquipment 系列控件。此控件的编程非常方便，用户可参照此例编程。下图是本例的中一个界面。



### 6.8 VCGraphDemo 例子

本例子主要是在VC境中怎样使用VARIANT类型赋值，主要用于Graph和Chart控件。用户可参照此例进行编程。下图是本例的中一个界面。

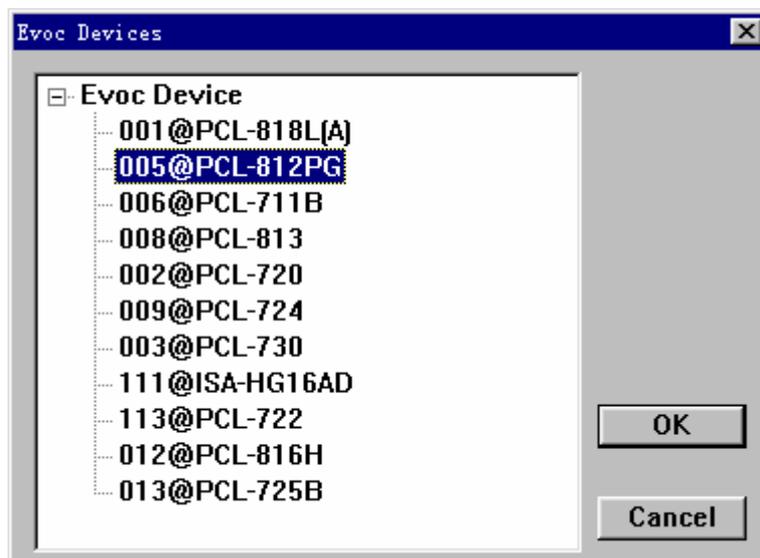


### 6.9 DASAI Demo 例子

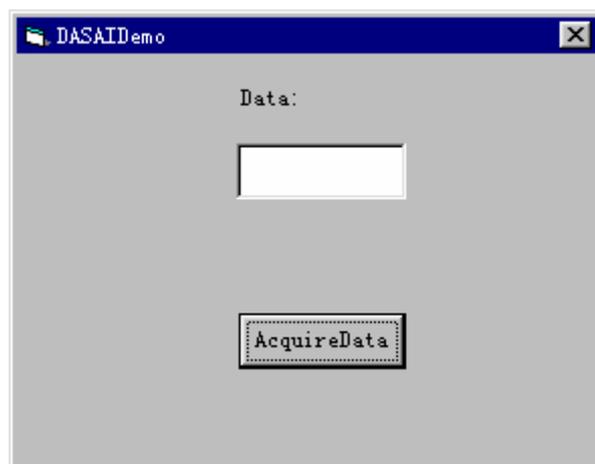
本例子用来使用 DASAI 控件来进行模拟量的采集输入。然后取得转化后的数字量。须至少插上一块本控件所支持的卡。具体可参考第七章。要求用户对该卡的硬件设置有基本的了解。然后用本软件包中的配置工具 DevInst.exe 对卡的资源进

行配置（包括端口、中断、DMA）。具体配置方法参考第八章。否则此例不能正常运行。

先如下图这样选择一个所支持的卡。

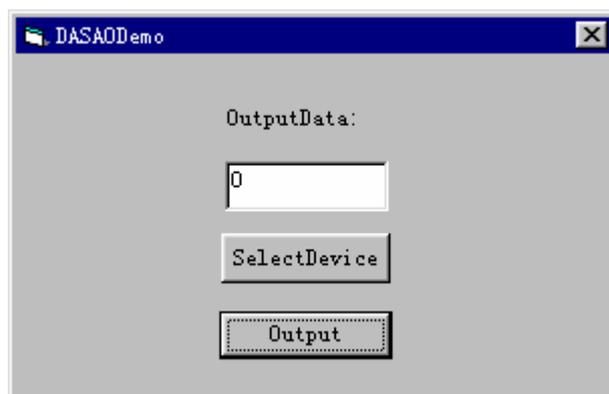


下图是本例中的一个界面。



#### 6.10 DASAoDemo 例子

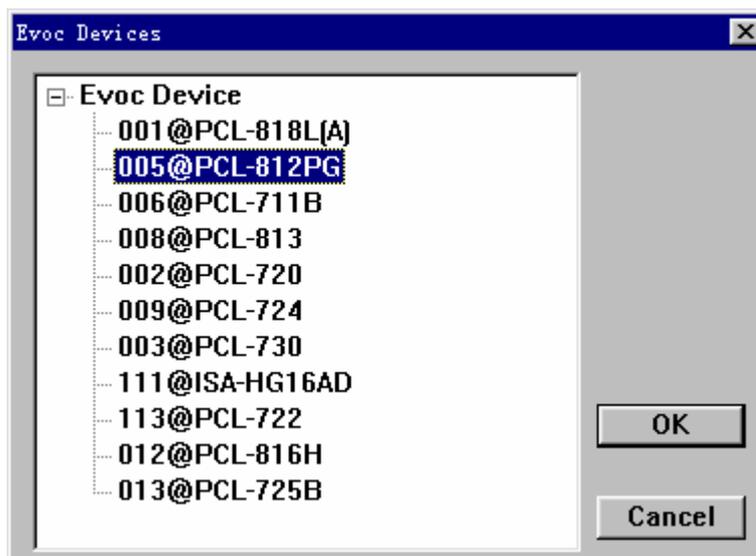
本例子用来使用DASAO控件来进行模拟量的输出。须至少插上一块本控件所支持的卡。具体可参考第七章。要求用户对该卡的硬件设置有基本的了解。然后用本软件包中的配置工具DevInst.exe对卡的资源进行配置（包括端口、中断、DMA）。具体配置方法参考第八章。下图是本例中的一个界面。



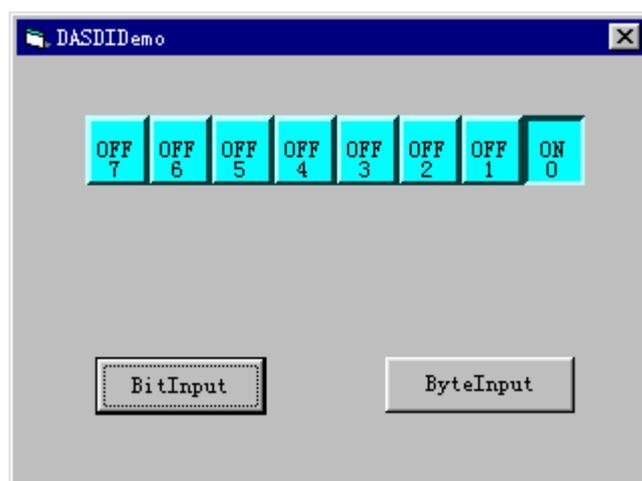
## 6.11 DASDiDemo 例子

本例子用来使用DAS D I 控件来进行形状开关量的输入。须至少插上一块本控件所支持的卡。具体可参考第七章。要求用户对该卡的硬件设置有基本的了解。然后用本软件包中的配置工具DevInst.exe对卡的资源进行配置（包括端口、中断、DMA）。具体配置方法参考第八章。下图是本例中的一个界面。

先如下图这样选择一个所支持的卡。



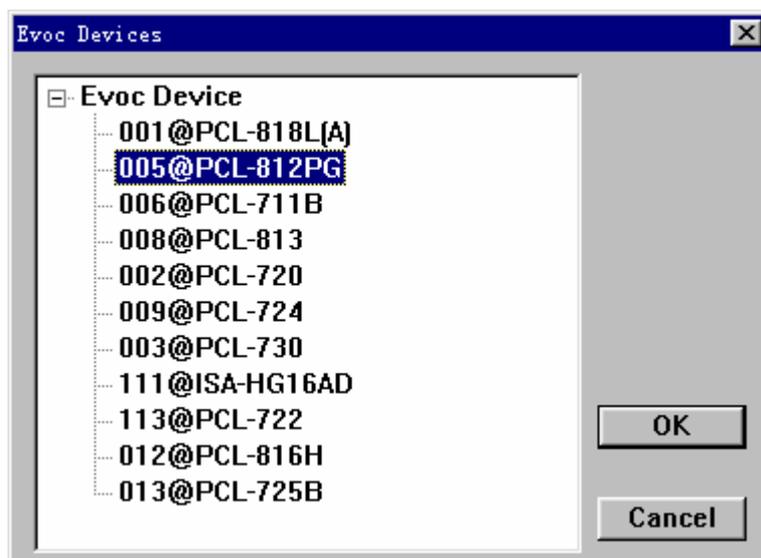
下图是本例中的一个界面。



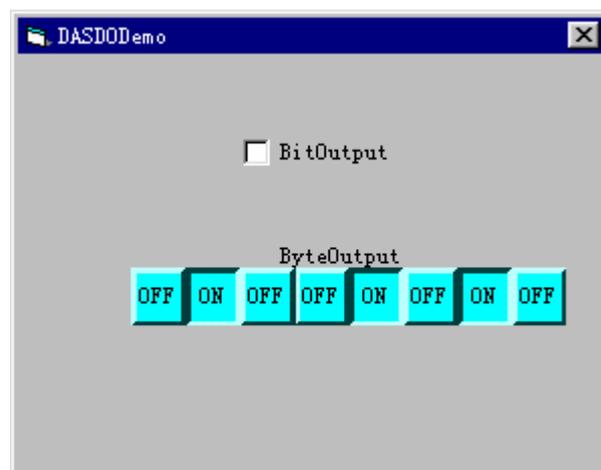
#### 6.12 DASDoDemo 例子

本例子用来使用DASDO 控件来进行形状开关量的输入。须至少插上一块本控件所支持的卡。具体可参考第七章。要求用户对该卡的硬件设置有基本的了解。然后用本软件包中的配置工具DevInst.exe对卡的资源进行配置（包括端口、中断、DMA）。具体配置方法参考第八章。下图是本例中的一个界面。

先如下图这样选择一个所支持的卡。



下图是本例中的中的一个界面。



### 6.13 其它注意事项

使用控件前要先注册MFC库，如：regsvr32

C:\windows\system\mfc42.dll

大部分控件的属性最好通过属性页设置，不要直接用程序编程。但Knob、Slide、E7Segment、Boolean控件的Value属性值可直接用编程方式设置。对于Knob和Slide控件，不要对其Value属性用程序赋值，只对其相应指针的Value赋值。再则，控件的方法可直接用编程方式调用。

另外，强烈建议屏幕分辨率在增强色(16位以上)。

## 第七章 硬件操作类控件支持的设备类型及其安装方法

### 7.1 各控件支持的本公司的卡的类型如下

- 1 . DASAI : PCL-711B , PCL-818L , PCL-813 , PCL-812PG ,  
PCL-816H , ISA-HG16AD
- 2 . DASAO : PCL-711B , PCL-818L , PCL-812PG , PCL-726 ,  
PCL-728 , ISA-HG16AD
- 3 . DASDI : PCL-720 , PCL-722 , PCL-724 , PCL-725 , PCL-725/16 ,  
PCL-730 , PCL-711B , PCL-818L ,  
PCL-812PG , PCL-816H , PCL-726 , ISA-HG16AD
- 4 . DASDO : PCL-720 , PCL-722 , PCL-724 , PCL-725 , PCL-725/16 ,  
PCL-730 , PCL-711B , PCL-818L ,  
PCL-812PG , PCL-816H , PCL-726 , ISA-HG16AD
- 5 . DASCOUNTER : PCL-720 , PCL-730 , PCL-818L , PCL812PG ,  
PCL-816H
- 6 . DASPulse : PCL-720 , PCL730 , PCL-818L , PCL-812PG ,  
PCL-816H
- 7 . PCL731A : PCL-731(A)
- 8 . PCL733 : PCL-733
- 9 . PCL734 : PCL-734
- 10 . PCL836A : PCL-836(A)

### 7.2 安装方法

安装您所使用的ISA卡的对应驱动程序，然后安装控件包

ActiveDAS。

Win98版：

请先安装您所使用的95&98版ISA卡的驱动程序，然后再安装ActiveDAS for Win98控件包。控件使用前，先用本软件包中的配置工具DevInst.exe对卡的资源进行配置（包括端口、中断、DMA）。具体配置方法参考第八章。使用控件前要先注册MFC库，如：regsvr32

C:\windows\system\mfc42.dll

1. DLL1 for Win95&98支持的卡：PCL-720, PCL-722,  
PCL-724, PCL-725,  
PCL-725/16, PCL-730.
2. DLL2 for Win95&98支持的卡：PCL-711B, PCL-813,  
PCL-816H, PCL-818L,  
PCL-818HG,PCL-812PG,  
PCL-726, PCL-728.
3. DLL8 for Win95&98支持的卡：PCL-816H
3. 73x for Win98支持的卡： PCL-731(A), PCL-733,  
PCL-734,
5. 836A for Win98&WinNT支持的卡：PCL-836(A)
6. ISA-HG16AD for 98支持的卡：ISA-HG16AD

WinNT版：

请先安装您所使用的 NT 版 ISA 卡的驱动程序，然后再安装ActiveDAS for WinNT控件包。板卡控件使用前，先用本软件包中的配置工具DevInst.exe对卡的资源进行配置（包括端口、中断、DMA）。具体配置方法参考第八章。使用控件前要先注册MFC库，如：regsvr32

C:\windows\system\mfc42.dll。

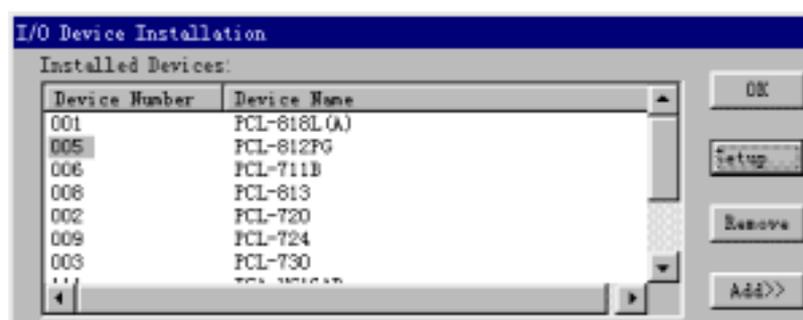
1. DLL1 for WinNT支持的卡：PCL-720, PCL-722, PCL-724, PCL-725, PCL-725/16, PCL-730.
  
2. DLL2 for WinNT支持的卡：PCL-711B, PCL-813, PCL-816H, PCL-818L, PCL-818HG, PCL-812PG, PCL-726, PCL-728.
  
3. DLL8 for WinNT支持的卡：PCL-816H
  
4. 836A for Win98&WinNT支持的卡：PCL-836(A)



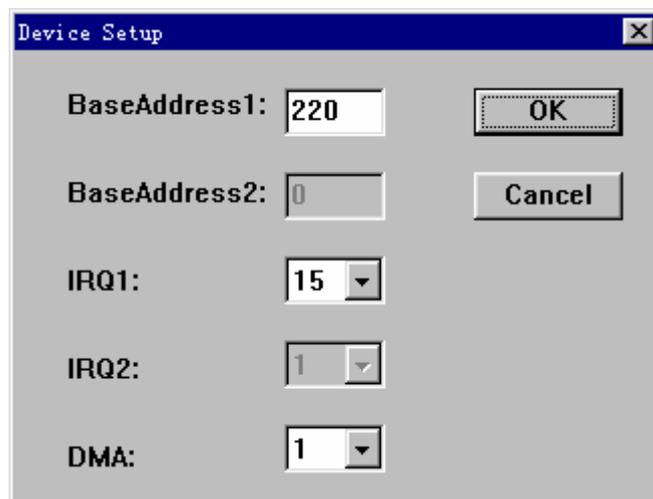
## 第八章 配置工具 DevInst.EXE 的使用

本配置工具专门用来对硬件操作类控件(目前本软件包只包括板卡驱动类控件)所要驱动的硬件的资源进行配置。特别是不具有即插即用功能的 I S A 卡,需要对它用到的资源(包括端口、中断、D M A )进行登记。这样,控件就会根据这些信息进行对硬件操作。在配置这些信息时,一定要与卡的跳线一致。即软件配置要与硬件设置一致,否则将不能正确地驱动硬件,甚至会使您的系统受到损坏。

具体哪些控件为硬件操作类控件以及支持卡的类型请参考第七章,会有详细的说明。配置完后,只要计算机硬件不作更改,此配置信息将一直有效。一旦卡的数目增加或减少,或者某卡的跳线作了变动,就要重新对其进行配置。已经配置好的硬件可通过本软件查看。下图就是程序刚启动起来的画面。显示在列表框中的硬件为本计算机的卡。当然,可能您的计算机上没插这么多卡。



用鼠标选中您计算机上存在的某卡的设备号,然后点击右部的 Setup (配置) 按钮。可看到下图中的提供的两个端口基地址、两个中断和两个 D M A , 有些卡用不着的资源将呈灰色不可编辑状态。框中编辑以十六进制为准。



如果你是第一次用本配置工具。列表框中将为空。那么您可点 Add 按钮添加新的设备。注意 DeviceNumber 与您所已经添加过的设备的 DeviceNumber 不能重复。点 Add it 按钮后, 此设备就加入了上面的列表框。然后再点 Setup 按钮对其进行配置即可。

