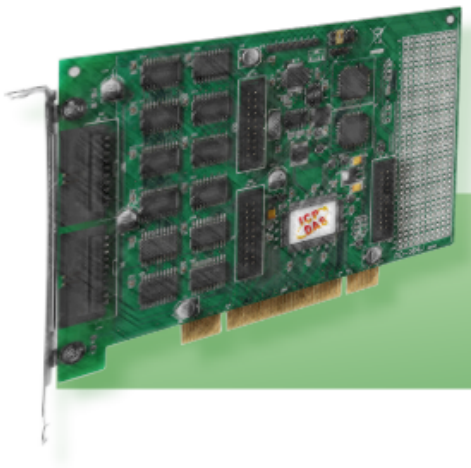


PIO-D64 系列

包含 **PIO-D64, PIO-D64U**

用户手册

简体中文版
版本: **1.3**
日期: **2011** 年 **11** 月

承诺

郑重承诺：凡泓格科技股份有限公司产品从购买即日起一年内无任何材料性缺损。

免责声明

凡使用本系列产品除产品质量所造成的损害，泓格科技股份有限公司不承担任何法律责任。泓格科技股份有限公司有义务提供本系列产品可靠而详尽数据，但保留修订权利，且不承担使用者非法利用数据对第三方所造成侵害构成的法律责任。

版权

版权所有 © 2011 泓格科技股份有限公司，保留所有权力。

商标

手册中所涉及所有公司商标，商标名称及产品名称分别属于该商标或名称的拥有者所有。

许可

用户可以使用、修改和备份单片机上的软件部份，但无权复制，转移或发布该软件的全部或部份拷贝。

目录

1 绪论

PIO-D64U 板卡是泓格新上市并符合 RoHS 环保规范的产品，新的 PIO-D64U 的设计可直接兼容于 PIO-D64。

PIO-D64 具有 32 路数字量输入信道和 32 路数字量输出信道。及 6 路定时器/计数器通道。我们也提供两块扩展板 DB-16P 和 DB-16R。PIO-D64 包含 2 路 16 位输入埠和 2 路 16 位输出埠.用户可以用 DB-16P 连接到输入埠(CN2,CN4)达到隔离的目的，或用 DB-16R 连接输出埠(CN1,CN3)来达到继电器输出的功能。板上具有 4 个时钟源：2M，1M，500K 和 250K.用户可以从焊脚上引用时钟源.板上的定时器/计数器提供 3 个信道用于频率测量、时间计数及产生脉冲。另一个 8254 提供 3 信道用于中断功能。

PIO-D64U 还有其它实用的功能, 第一种是 Card ID 指拨开关, 让使用者可以自由设定每张板卡的识别码.当系统同时使用多张 PIO-D64U 板卡时,使用者可以迅速而简单区别这些同型号的板卡。第二种是 DI Pull High/Low 设定功能, 数字输入端口可设定为 pull-high 或 pull-low ，当信号线脱落或断线时，该 DI 值会相对维持 High 或 Low 的状态(非浮动)。

此系列卡支持在 Linux、DOS、Windows 98/NT/2000、32/64-Bit Windows XP/2003/Vista/2008/7 等操作系统环境下使用，还提供有动态函式库及 Active X 控件使开发更加容易及简单易懂的各种语言范例程序，如 Turbo C++、Borland C++、Microsoft C++、Visual C++、Borland Delphi、Borland C++ Builder、Visual Basic、Visual C#.NET、Visual Basic.NET 及 LabVIEW 等，让用户能够快速的上手来使用。

1.1 规格

PIO-D64/PIO-D64U

板卡名称	PIO-D64	PIO-D64U
数字输入		
通道数	32	
兼容性	5 V/TTL	
输入电压	Logic 0: 0.8 V max. Logic 1: 2.0 V min.	
响应速度	1 MHz	
数字输出		
通道数	32	
兼容性	5 V/TTL	
输出电压	Logic 0: 0.4 V max. Logic 1: 2.4 V min.	
输出能力	Sink: 24 mA @ 0.8 V Source: 15 mA @ 2.0 V	
响应速度	1 MHz	
定时器/计数器		
通道数	6 (Independent x3, EVTIRQ x1, EXTIRQ x1)	
分辨率	16-bit	
兼容性	5 V/TTL	
参考时钟	Internal: 4 MHz	
公共		
总线型态	5 V PCI, 32-bit, 33 MHz	3.3 V/5 V Universal PCI, 32-bit, 33 MHz
数据总线	8-bit	
卡 ID	无	有(4-bit)
I/O 连接头	20-pin 接头 x 5	
尺寸(长 x 宽 x 高)	156 mm x 110 mm x 22 mm	
耗电量	580 mA @ +5 V	
运行温度	0 ~ 60 °C	
储存温度	-20 ~ 70 °C	
周围环境相对湿度	5 ~ 85% 相对湿度, 非冷凝(non-condensing)	

1.2特点

- PIO-D64 为 PCI bus 接口，支持 +5 V PCI bus 插槽
- PIO-D64U 为 Universal PCI 接口，支持 +3.3 V 及+5 V PCI bus 插槽
- 32 个数字量输出信道
- 32 个数字量输出信道
- 4 个独立可编程的 16 位计时/计数器
- 一个 32 位的内建程序计数器为 4 MHz Clock
- 提供 4 个时钟源：2M , 1M , 500K 和 250K。
- 提供 3 个中断源通道
- Card ID 功能为 PIO-D64U 仅有
- 数字输入端口可设定为 Pull-high 或 Pull-low 为 PIO-D64U 仅有
- 短卡设计
- 可直接连接 DB-24PR, DB-24PRD, DB-24POR, DB-24C, DB-16P, DB-16R 或者其它兼容 OPTO-22 规格的端子板

1.3 检查产品清单

硬纸盒包装内包括以下项目：

- 一张 PIO-D64/D64U 系列板卡
- 一张 软件安装的 PCI 光盘。
- 一张 快速入门指南。

建议您先阅读快速入门指南。此快速入门指南，包括所有必要和重要的信息如下：

- 从哪里获得软件驱动程序、演示程序和其它资源。
- 如何安装软件。
- 如何测试使用板卡。

注意：

如发现产品包装内的配件有任何损坏或遗失，请保留完整包装盒及配件，尽快联系我们，我们将有专人快速为您服务。

联络方式(E-Mail): **service@icpdas.com , service.icpdas@gmail.com**

2 硬件结构

2.1 板卡布局

■ PIO-D64/D64U 板卡布局

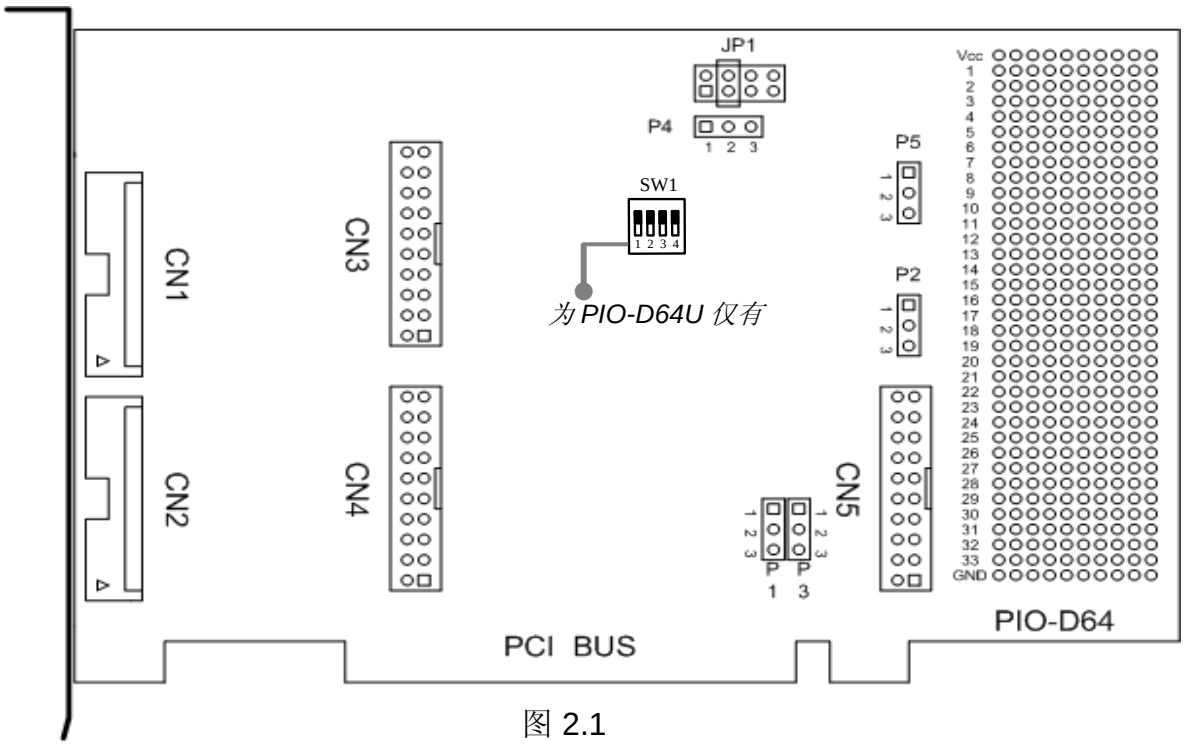


图 2.1

注意:

CN1	数字量输出 0~15 通道
CN2	数字量输入 0~15 通道
CN3	数字量输出 16~31 通道
CN4	数字量输入 16~31 通道
CN5	计时/计数功能 0~4 信道
JP1	4 个时钟源 2M , 1M , 500K 和 250K
SW1	Card ID 功能为 PIO-D64U 仅有

2.2 I/O 口位置

PIO-D64 系列卡具有 2 个 16 位数字量输入端口和 2 个 16 位数字量输出端口。这些 I/O 口配置如下：

I/O 口配置说明：

表 2.1

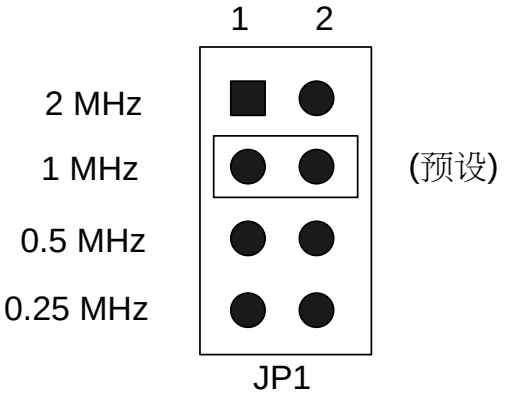
板卡名称/连接头		说明
PIO-D64 PIO-D64U	CN1	DO0 ~ DO15
	CN2	DI0 ~ DI15
	CN3	DO16 ~ DO31
	CN4	DI16 ~ DI31

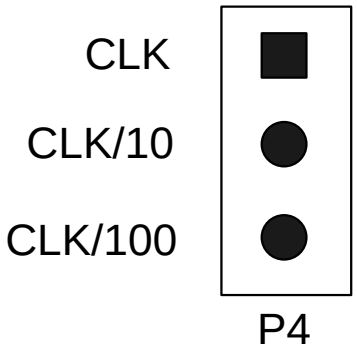
板卡名称/连接头		说明
PIO-D64 PIO-D64U	CN5	Timer/Counter 0~4

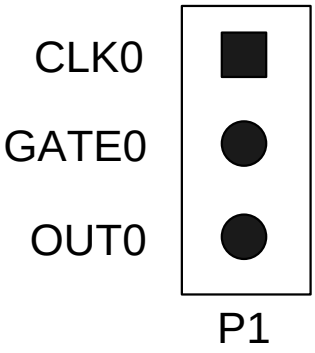
注意 1：参考章节 2.1 **板卡布局及 I/O 配置。**

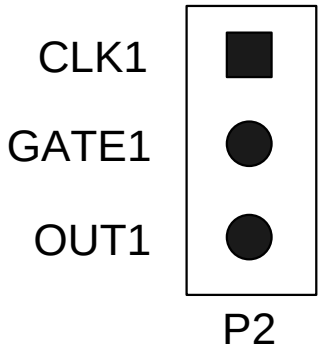
2.3 JP1 时钟源

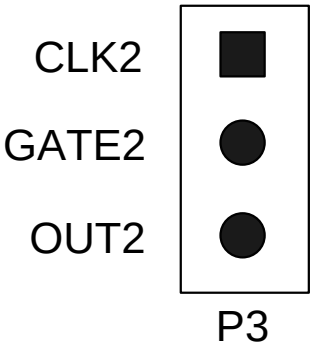
PIO-D64 系列板卡提供了可透过 JP1 跳接器来选择 4 个时钟源为 2 MHz, 1 MHz, 500 KHz 和 250 KHz。

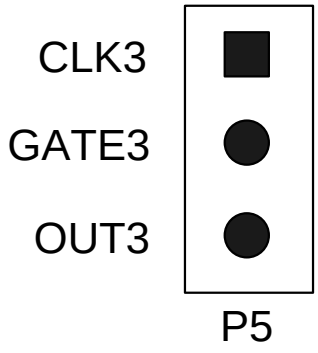






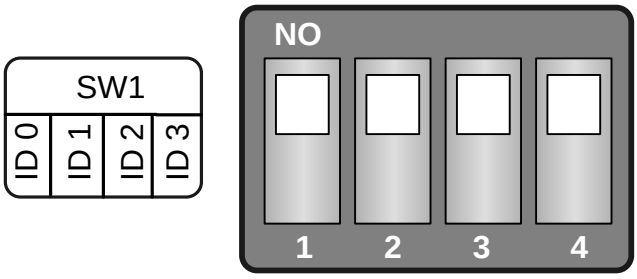






2.4 Card ID 开关

PIO-D64U 在硬件上新增 Card ID 指拨开关, 此功能为 PIO-D64U 仅有，让使用者可以自由设定每张板卡的识别码。当系统同时使用多张 PIO-D64U 卡时，使用者可以迅速而简单区别这些同型号的板卡。出厂预设 Card ID 为 0x0。详细的 SW1 Card ID 设定，请参考至表 2.2。



(预设设定)

表 2.2 (*) 预设设定; OFF → 1; ON → 0

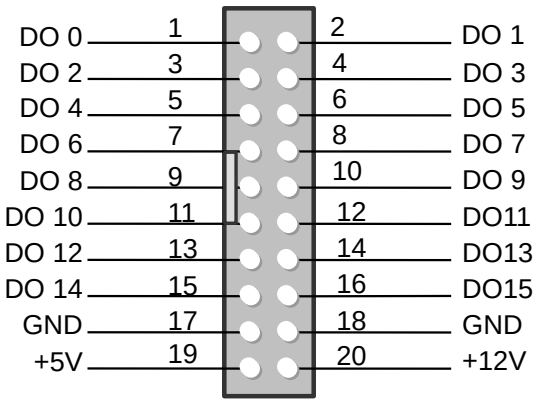
Card ID (Hex)	1 ID0	2 ID1	3 ID2	4 ID3
(*) 0x0	ON	ON	ON	ON
0x1	**OFF**	ON	ON	ON
0x2	ON	**OFF**	ON	ON
0x3	**OFF**	**OFF**	ON	ON
0x4	ON	ON	**OFF**	ON
0x5	**OFF**	ON	**OFF**	ON
0x6	ON	**OFF**	**OFF**	ON
0x7	**OFF**	**OFF**	**OFF**	ON
0x8	ON	ON	ON	**OFF**
0x9	**OFF**	ON	ON	**OFF**
0xA	ON	**OFF**	ON	**OFF**
0xB	**OFF**	**OFF**	ON	**OFF**
0xC	ON	ON	**OFF**	**OFF**
0xD	**OFF**	ON	**OFF**	**OFF**
0xE	ON	**OFF**	**OFF**	**OFF**
0xF	**OFF**	**OFF**	**OFF**	**OFF**

2.5 引脚分配

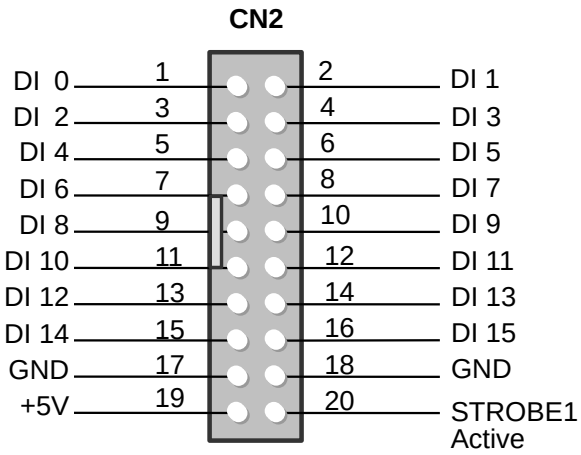
PIO-D64 连接器引脚分配参考至下列。所有数字量输入或数字量输出信号均为 TTL 电平相兼容。

■ **CN1~CN4: 20 针扁平连接器 (数字量输入/数字量输出)**

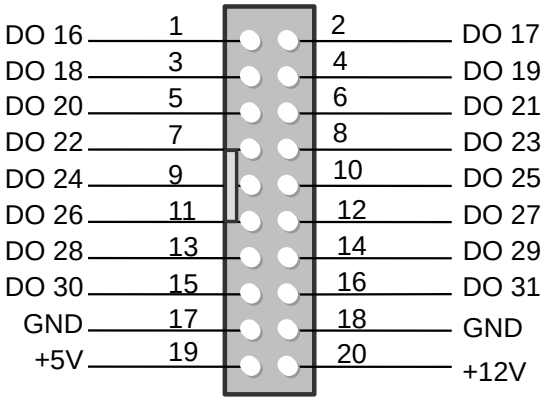
CN1



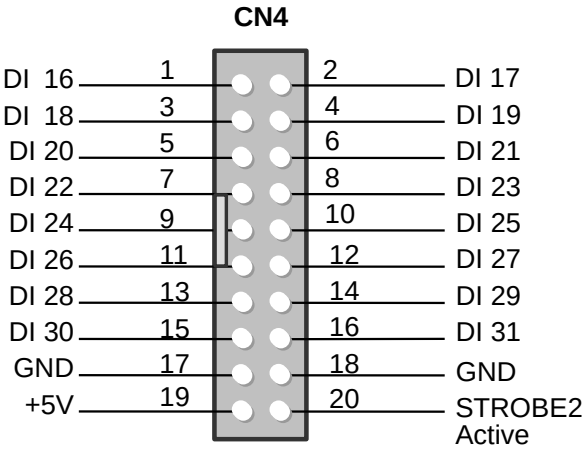
CN2



CN3

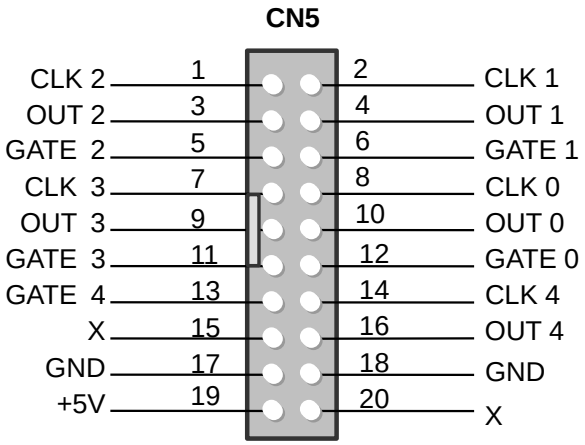


CN4



■ **CN5: 20 针扁平连接器 (计时/计数功能)**

CN5



2.6 I/O 操作

2.6.1 D/O 端口结构 (CN1 及 CN3)

当 PC 上电后，RESET\ 信号将所有的 DO 状态清除到低电平状态。更多关于 RESET\信号可以参考章节 3.3.1。注：DO 功能详细说明见图 2.2。

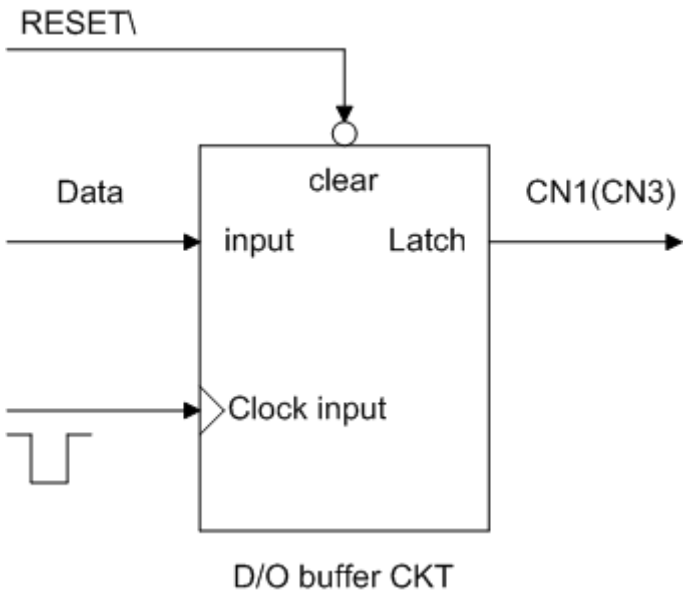


图 2.2: D/O 功能图块

2.6.2 D/I 端口结构(CN2 及 CN4)

D/I 端口被 RESET\信号控制到激活/禁用，具体描述如下：

- RESET\为 Low-state → 所有 DI 运行状态是禁用
- RESET\ 为 High-state → 所有 DI 运行状态是激活

当 PC 上电后，所有运行的 D/I 端口为禁用因此 RESET\ 为低电平。此外，在一些应用中用户可能需要拴锁外部输入数据，我们提供下列结构，见图 2.3，允许用户应用 STROBE 针脚去拴锁 D/I 输入信号。**如果没有信号拴锁 STROBE 针脚，那么输入数据会直接进到数字量输入寄存器。**

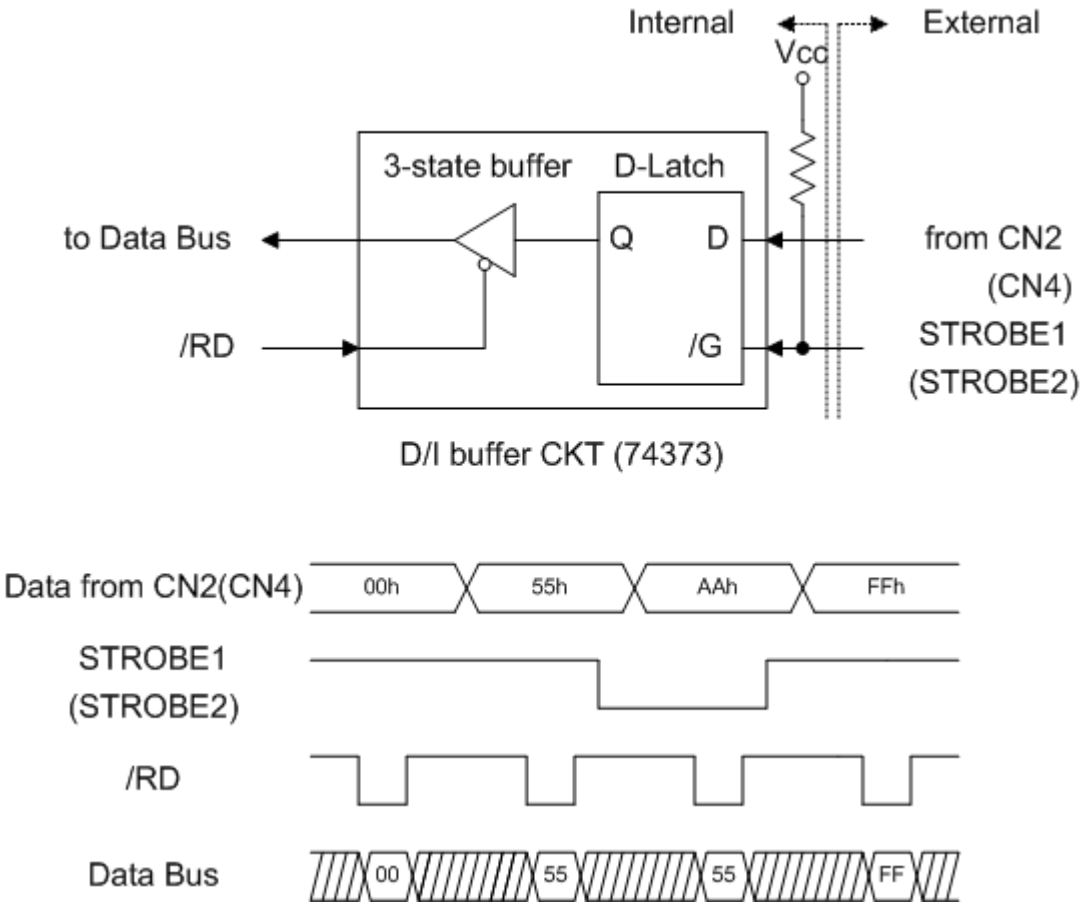


图 2.3

2.7 Timer/Counter 结构

PIO-D64 有 2 个定时器/计数器芯片，8254。第 1 个 8254 芯片是被使用在一般情况下的 timer/counter，见图 2.4。针脚分配说明在章节.2.5。

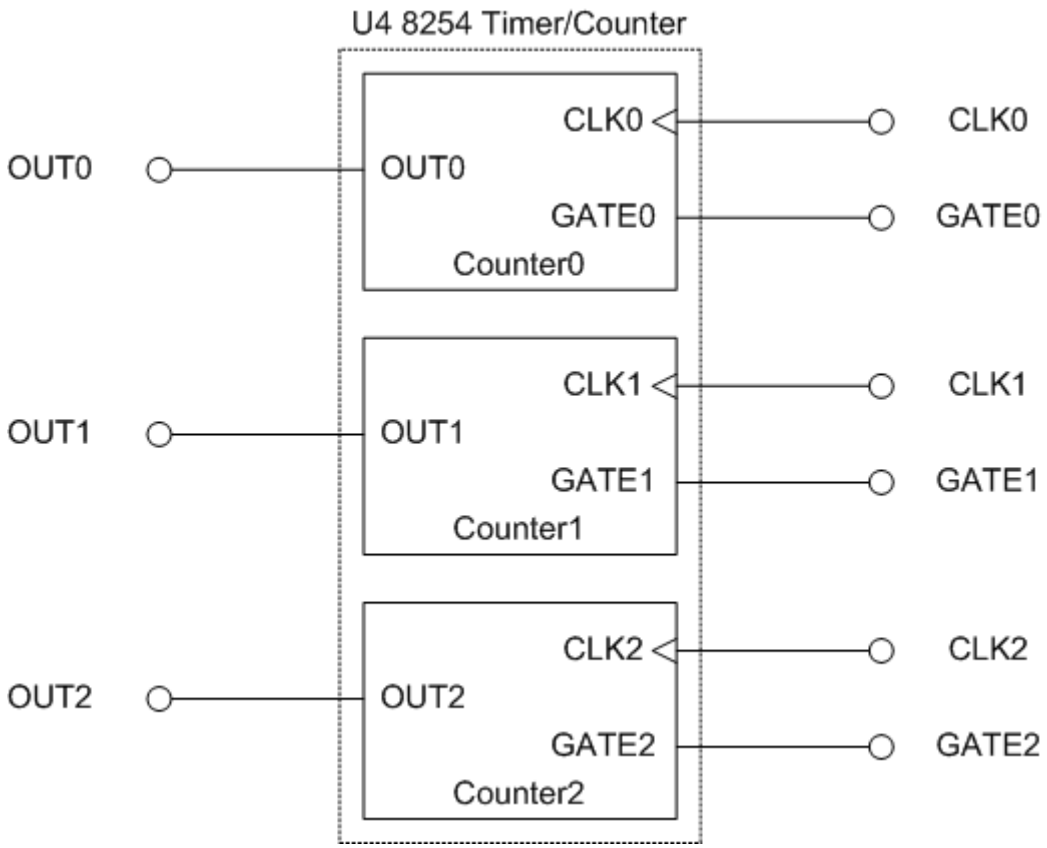


图 2.4

第 2 个 8254 芯片是被使用在产生中断触发信号，见图 2.5。Counter3 接受信号事件并将产生中断触发信号。Counter4 和 Counter5 是层叠形式，时钟源是 4 MHz，被使用来定时产生中断信号。

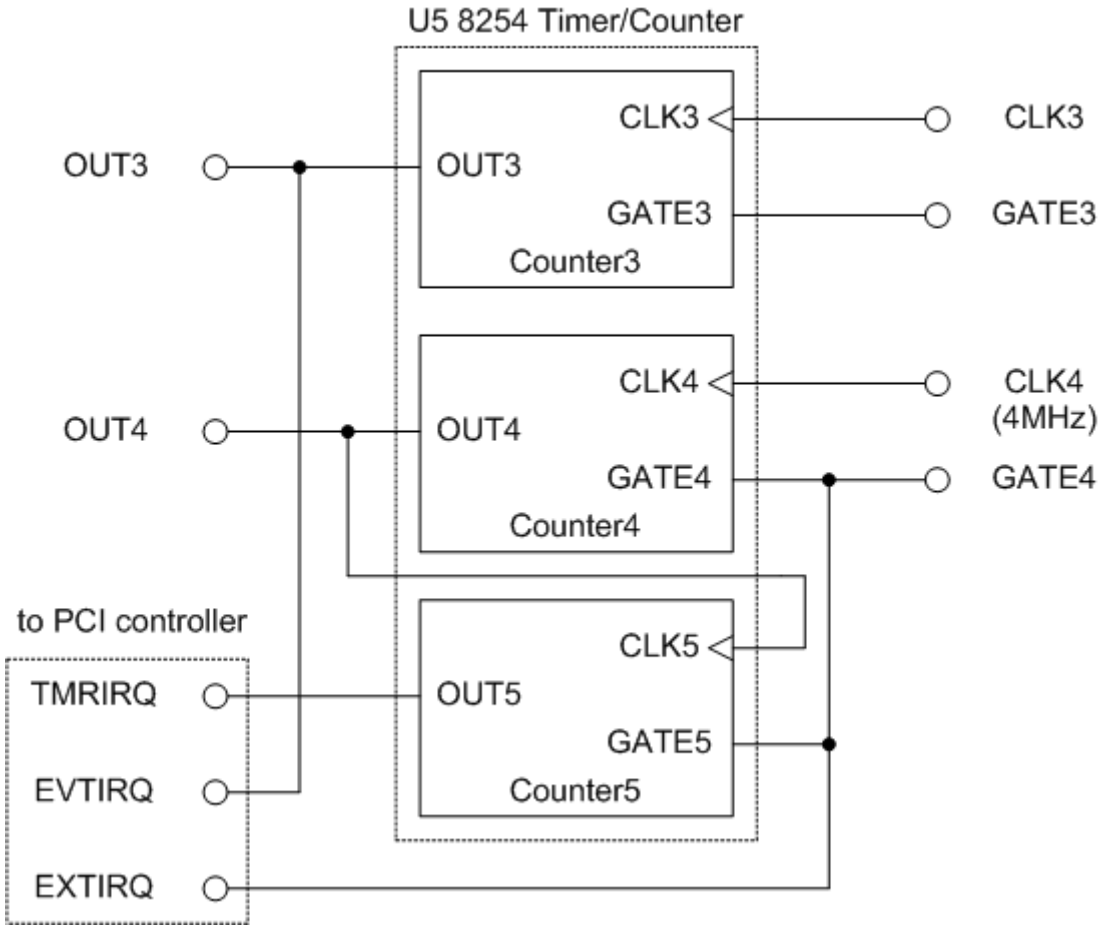
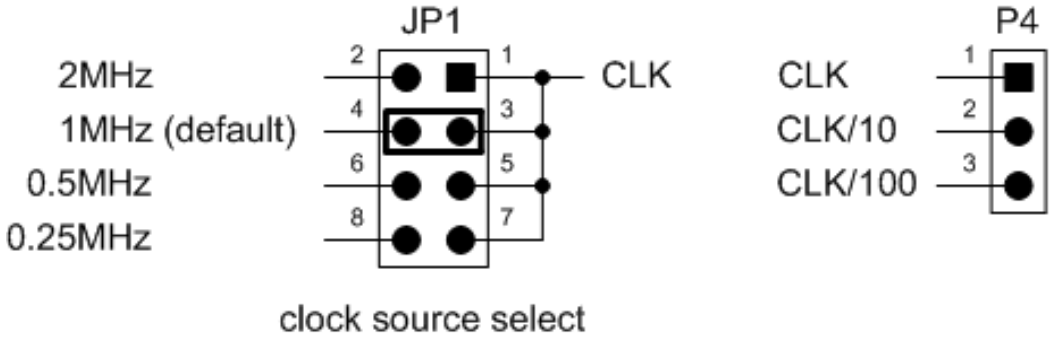


图 2.5

注意：更多关于引脚定义信息参考章节.2.5
更多关于中断运行方式信息参考章节.2.9

2.8 时钟源

在下表中 PIO-D64 系列板卡提供了多种范围的时钟源。跳线设置 JP1，在 P4 接点上用户能选择适当的时钟输出。



clock source select

JP1 setting	P4 soldering pad clock output		
	P4.1	P4.2	P4.3
1-2	2MHz	200KHz	20KHz
3-4 (default)	1MHz	100KHz	10KHz
5-6	500KHz	50KHz	5KHz
7-8	250KHz	25KHz	2.5KHz

2.9 中断运行

在 PIO-D64 系列板卡中有 3 个中断源。这 3 个信号被指定为 INT_CHAN_0，INT_CHAN_1，INT_CHAN_2。它们的信号源参考：

INT_CHAN_0: EXTIRQ
INT_CHAN_1: EVTIRQ
INT_CHAN_2: TMRIRQ

如果只有 1 个中断信号源被使用，那么中断服务程序可以不去识别中断源。更多信息参考 DOS 系统的 DEMO3.C，DEMO4.C，DEMO5.C。

如果超过 1 个中断源，中断服务程序得去确定激活信号。更多信息参考 DOS 系统的 DEMO6.C。

1. 读取所有中断信号源最新状态
2. 比较 new status 和 old status 去识别激活信号
3. 如果是 INT_CHAN_0 激活, 执行它的中断服务程序
4. 如果是 INT_CHAN_1 激活, 执行它的中断服务程序
5. 如果是 INT_CHAN_2 激活, 执行它的中断服务程序
6. 更新中断状态

如果中断信号太短，则状态保持原状态不会改变。中断服务程序不能确定在这个条件下中断源能够激活。因此在中断服务程序执行前中断信号需要长时间 hold_active。不同的操作系统是不同的 Hold_time。Hold_time 的范围由毫秒到秒。一般情况下操作系统 20 ms 足够。

2.9.1 中断功能块图表

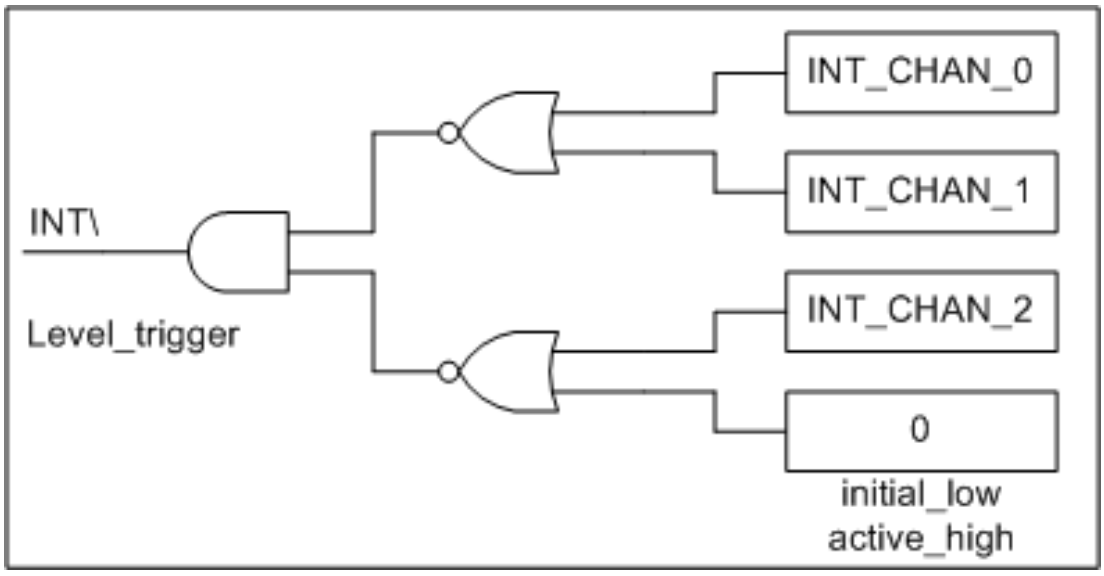


图 2.6

PIO-D64 系列卡中断输出信号，INT\中断信号是低电平触发。如果 INT\产生一个低平脉冲，PIO-D64 系列卡将中断 PC time 一次。如果 INT\固定在低平信号，PIO-D64 系列卡将连续中断 PC。所以，正规请求，INT_CHAN_0/1/2 需要被一个脉冲类型的信号约束。也就是说，他们通常需要被固定低平信号并产生一个 high_pulse 去中断 PC。

INT_CHAN_0/1/2 优先权一样。如果这 3 个信号是同时激活。那么 INT\将激活一次 time。因此中断服务程序去读取所有中断通道状态。参考 DOS 系统环境的 DEMO6.C。这个示例是 INT_CHAN_1 and INT_CHAN_2 条件下的应用。

如果只使用一个中断源，中断服务程序没有去读取中断源状态。DEMO 程序，DEMO3.C，DEMO4.C，DEMO5.C 是在 DOS 环境下单信道中断应用程序：

DEMO3.C → INT_CHAN_0
DEMO4.C → INT_CHAN_1
DEMO5.C → INT_CHAN_2

2.9.2 INT_CHAN_0 (外部中断的控制)

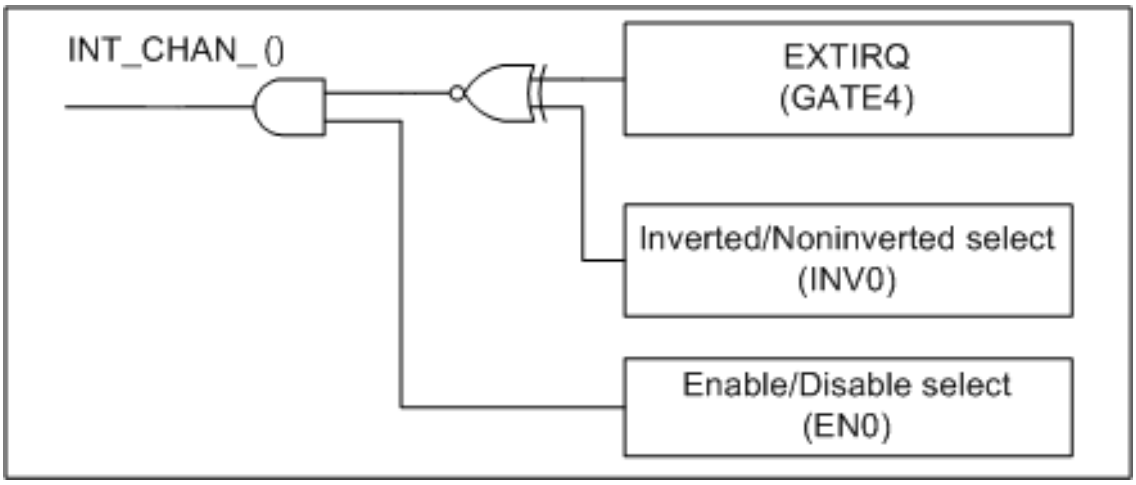


图 2.7

图 2.7 是对于外部中断的控制方法。信号源来自 GATE4。INV0 决定是否要将触发信号源反向，EN0 是使来 禁用/激活 INT_CHAN_0（Pin13 of CN5）（参考章节 2.7 中断信号源）。**正常 INT_CHAN_0 必须被固定低平状态和一个 high_pulse 去中断 PC。**

1. EN0 能指定 INT_CHAN_0 是要激活或是禁用 (参考图 2.7)
   - EN0=0 → INT_CHAN_0=禁用
   - EN0=1 → INT_CHAN_0=激活

2. EXTIRQ 的 INV0 能决定是不是要反向外部中断信号源 (参考图 2.7)
   - INV0=0 → INT_CHAN_0=反向状态为 EXTIRQ
   - INV0=1 → INT_CHAN_0=正向状态为 EXTIRQ

注意：更多信息参考 DOS 系统 DEMO3.C

2.9.3 INT_CHAN_1 (事件中断的控制)

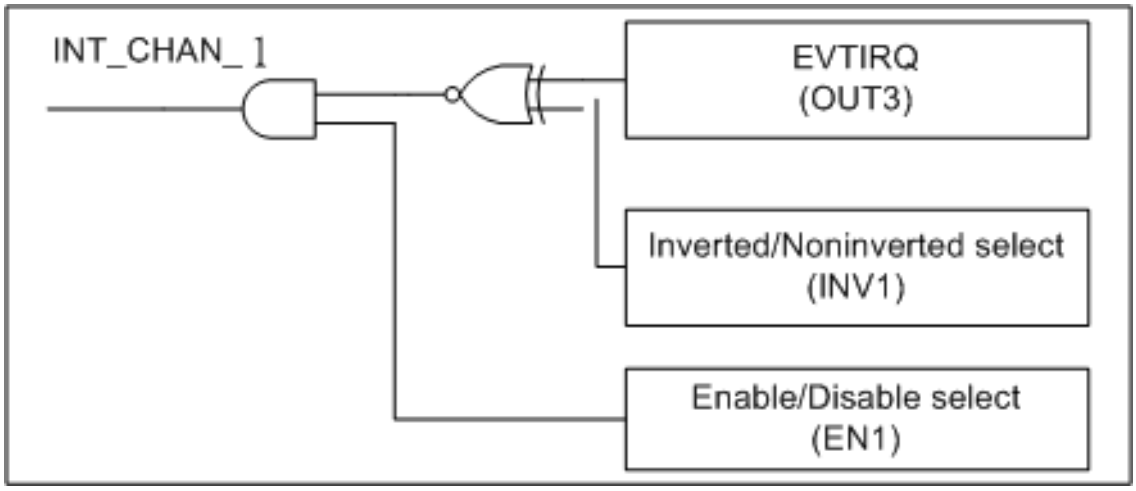


图 2.8

图 2.8 事件中断的控制方法，信号源来自 OUT3。INV1 决定是否要反向触发信号源，EN1 决定要激活还是禁用事件中断（参考章节 2.7 为中断信号源）。用户能使用 Counter3 事件计数器去计 CN5 Pin7 的事件信号。同时去设置 counter3，那么中断 INT_CHAN_1 将被触发。INT_CHAN_1 需要被固定在低电平状态正常，产生一个高平脉冲去中断 PC。

1. EN1 能指定 INT_CHAN_1 是要激活或是禁用 (参考图 2.8)
   - EN1=0 → INT_CHAN_1=禁用
   - EN1=1 → INT_CHAN_1=激活

2. EVTIRQ 的 INV1 能决定是不是要反向事件中断信号源 (参考图 2.8)
   - INV1=0 → INT_CHAN_1=反向状态为 EVTIRQ
   - INV1=1 → INT_CHAN_1=正向状态为 EVTIRQ

注意：更多信息参考 DOS 系统 DEMO4.C

2.9.4 INT_CHAN_2 (计时器中断的控制)

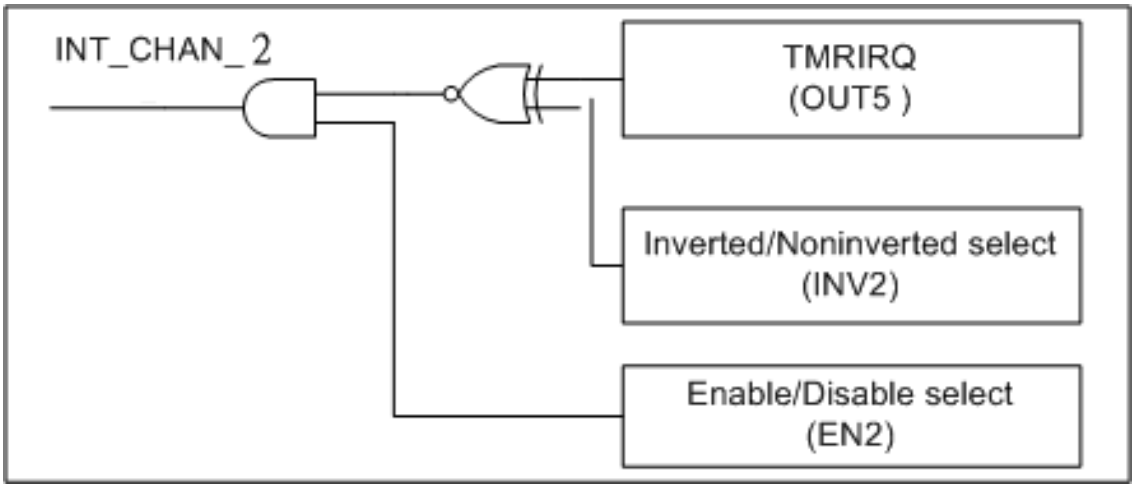


图 2.9

图 2.9 计时器中断控制方法，信号源来自 OUT5。INV2 决定是否要反向触发信号源，EN2 决定要激活还是禁用计时器中断。**注：INT_CHAN_2 必须被固定的低平信号和一个高平脉冲去中断 PC。**Counter4 和 Counter5 为层叠形式，它被使用成为一个 4 MHz 时钟源的 32-bit 中断计时器。

1. EN2 能指定 INT_CHAN_2 是要激活或是禁用 (参考图 2.9)
   - EN2=0 → INT_CHAN_2=禁用
   - EN2=1 → INT_CHAN_2=激活

2. TMRIRQ 的 INV2 能决定是不是要反向事件中断信号源 (参考图 2.9)
   - INV2=0 → INT_CHAN_2=反向状态为 TMRIRQ
   - INV2=1 → INT_CHAN_2=正向状态为 TMRIRQ

注意：更多信息参考 DOS 系统 DEMO5.C

2.10 端子板

2.10.1 DB-16P 光电隔离数字量输入板

DB-16P 是 16 通道光电隔离数字量输入板。DB-16P 光电隔离是由 1 个双向的光电偶隔离器和 1 个电流检测电阻组成的。你能使用 DB-16P 去检测从 TTL 电平到 24V 的直流信号，也可以用来检测宽范围的交流信号。你能使用它去隔离计算机和工业环境中常发生的共模电压，地环流以及电压尖峰。详细功能块见图 2.10。

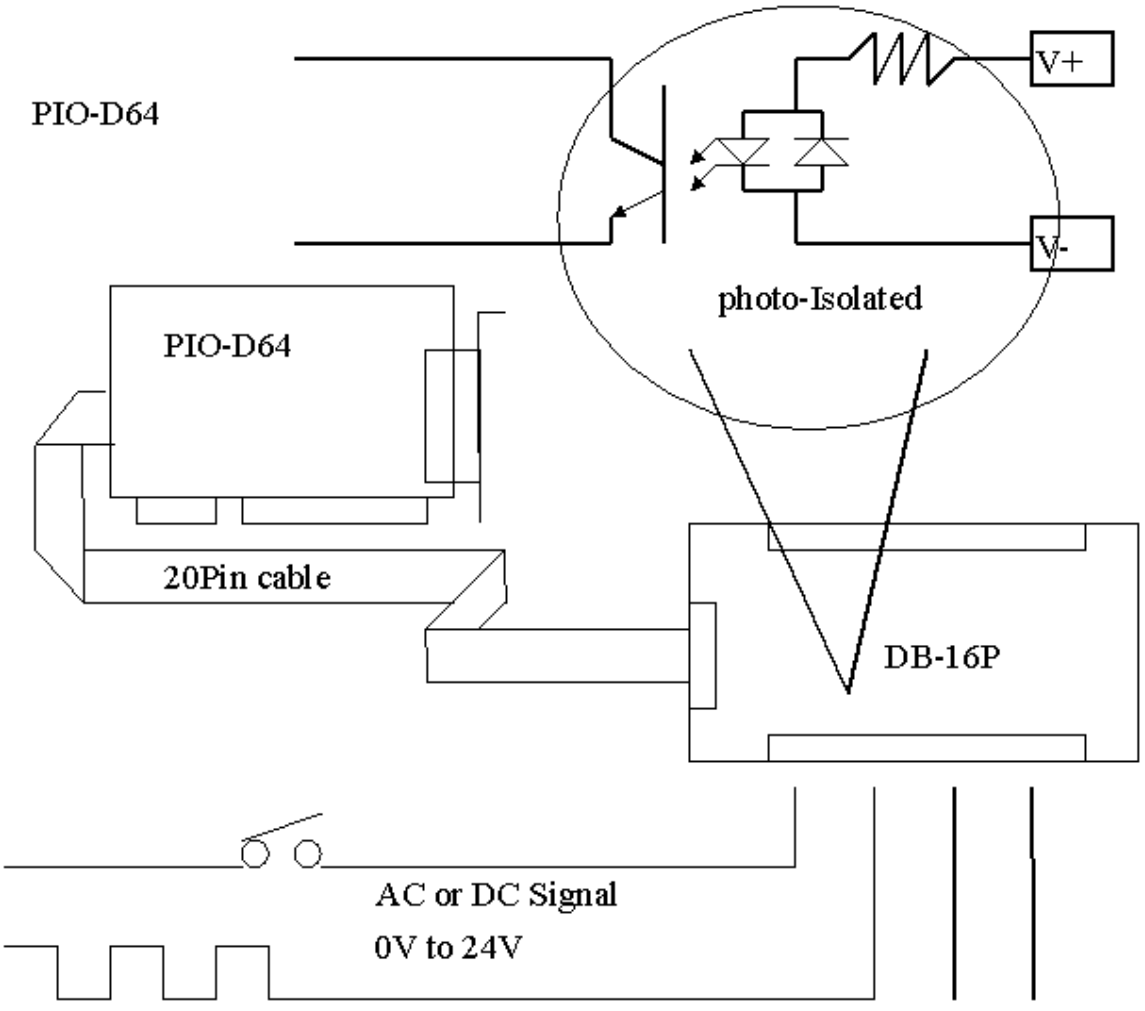


图 2.10

2.10.2 DB-16R 继电器输出板

DB-16R 是 1 个 16 通道继电器输出板，由 16 个 C 型继电器组成，可以用编程去控制。适合 12V/24V 电压信号，可以通过适当的 20-pin 扁平连接头连接。16 个 LED 去显示状态。功能块说明见图 2.11。

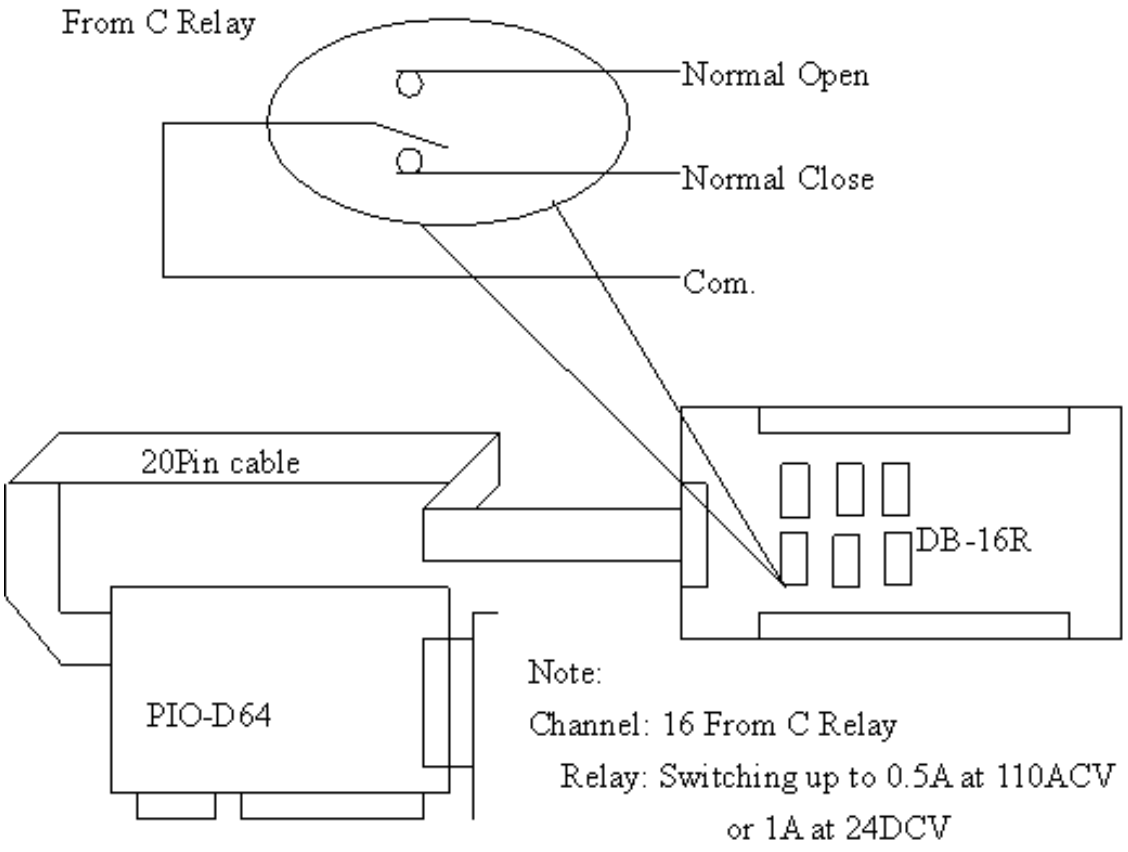


图 2.11

2.10.3 DB-24PRD, DB-24POR, DB-24C

表 2.3

名称	说明
DB-24PR	24 x Poer rely, 5 A/250 V
DB-24POR	24 x PhotoMOS relay, 0.1 A/350 V_{AC}
DB-24C	24 x open collector, 100 mA per channel, 30 V max.

DB-24PR 是 24 通道功率继电器输出板，是由 8 个 C 型继电器和 16 个 A 型继电器组成可以编程控制。每个继电器可以控制一个 25V_{AC} 或 30V_{DC} 的 5A 负载。通过 50-pin 与 OPTO-22 兼容的连接器或 20-pin 扁平电缆连接器，给适当通道的功率继电器加上 5V 电压信号来激活。每个通道都有一个 LED 指示，共有 24 个 LED 显示状态。本板卡需要一个+12V_{DC} 或+24V_{DC} 的电源供应。详细功能块说明见图 2.12。

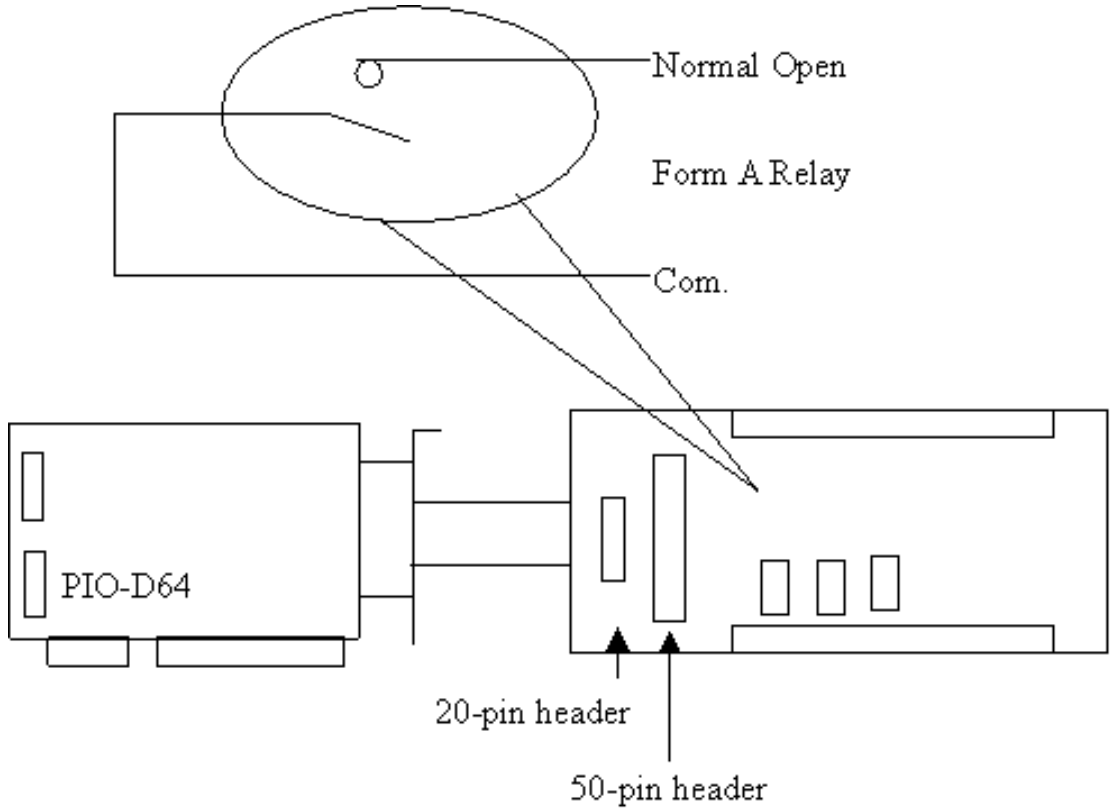


图 2.12

注意：50-Pin 连接器 (OPTO-22 兼容), 适用于 DIO-24, DIO-48, DIO-144, PIO-D144, PIO-D96, PIO-D56, PIO-D48, PIO-D24, PIO-D168(A)
通道: 16 个 A 型继电器, 8 个 C 型继电器
继电器: 开关为 5A / 110V_{AC} 或 5A / 30V_{DC}

2.10.4 端子板对照表

表 **2.4**

	20-pin flat-cable	50-pin flat-cable	D-sub 37-pin
DB-37	No	No	**Yes**
DN-37	No	No	**Yes**
ADP-37/PCI	No	**Yes**	**Yes**
ADP-50/PCI	No	**Yes**	No
DB-24P	No	**Yes**	No
DB-24PD	No	**Yes**	**Yes**
DB-16P8R	No	**Yes**	**Yes**
DB-24R	No	**Yes**	No
DB-24RD	No	**Yes**	**Yes**
DB-24C	**Yes**	**Yes**	**Yes**
DB-24PR	**Yes**	**Yes**	No
DB-24PRD	No	**Yes**	**Yes**
DB-24POR	**Yes**	**Yes**	**Yes**
DB-24SSR	No	**Yes**	**Yes**

注意：**PIO-D64** 系列卡仅适用 **20** 针扁平电缆。

3 I/O 控制寄存器

3.1 如何找到 I/O 地址

在上电后即插即用 BIOS 将分配适当的一个 I/O 地址到每个 PIO/PISO 板卡。PIO-D64 系列板卡 ID 如下:

PIO-D64/PIO-D64U			
Rev 1.0		**Rev 2.0 或更新版本**	
Vendor ID	0xE159	Vendor ID	0xE159
Device ID	0x0002	Device ID	0x0001
Sub-vendor ID	0x80	Sub-vendor ID	0x4080
Sub-device ID	0x01	Sub-device ID	0x01
Sub-aux ID	0x20	Sub-aux ID	0x20

工具程序 PIO_PISO.EXE，将找到并显示 PIO/PISO 卡安装在 PC 上的所有信息，此外，告诉你去识别 ICPDAS 数据采集卡的 PIO 卡，sub-vender, sub-device 和 sub-Aux ID，具体参数参考表 3-1。

PIO_PISO.exe 工具程序下载位置:

CD:\NAPDOS\PCI\Utility\Win32\PIO_PISO\

http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/utility/win32/pio_piso/

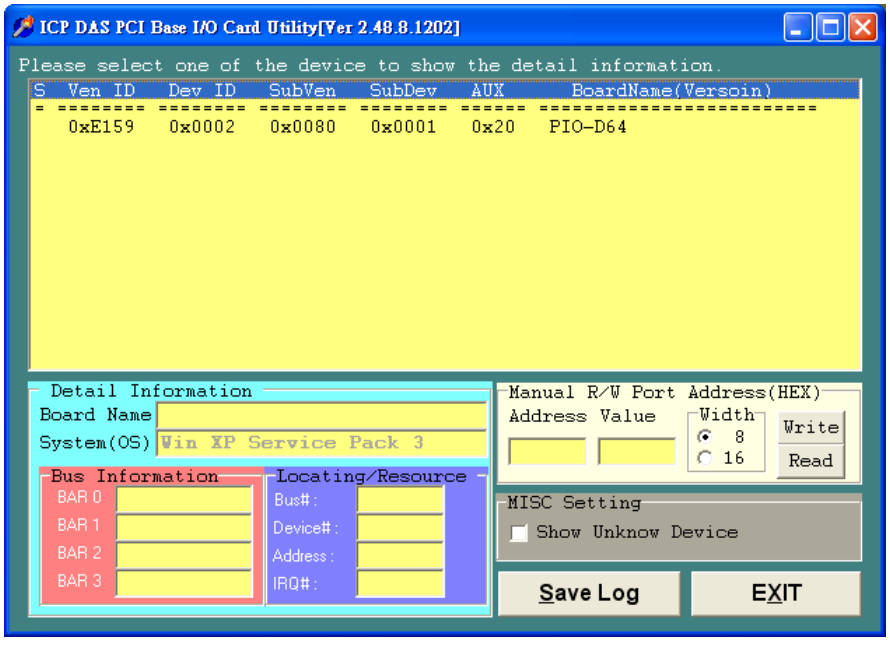


图 3.1

表 **3-1**

PIO/PISO 系列卡	说明	Sub_Vendor	Sub_Device	Sub_AUX
PIO-D168	168 * DIO	9880	01	50
PIO-D168A	168 * DIO	80	01	50
PIO-D144(REV4.0)	144 * D/I/O	80 (5C80)	01	00
PIO-D96 (REV4.0)	96 * D/I/O	80 (5880)	01	10
PIO-D64 (REV2.0)	64 * D/I/O	80 (4080)	01	20
PIO-D56	24 * D/I/O + 16 * D/I+16*D/O	80 (8080)	01	40
PIO-D48	48 * D/I/O	80 (0080)	01	30
PIO-D24	24 * D/I/O	80 (8080)	01	40
PIO-821	Multi-function	80	03	10
PIO-DA16	16 * D/A	80	04	00
PIO-DA8	8 * D/A	80	04	00
PIO-DA4	4 * D/A	80	04	00
PISO-C64	64 * isolated D/O (**Current sinking**)	80	08	00
PISO-A64	64 * isolated D/O **(Current sourcing)**	80	08	50
PISO-P64	64 * isolated D/I	80	08	10
PISO-P32C32	32* isolated D/O (**Current sinking**) + 32* isolated D/I	80(4280)	08(00)	20(00)
PISO-P32A32	32*isolated DO (**Current sourcing**) + 32* isolated D/I	80(C280)	08	70
PISO-P8R8	8* isolated D/I + 8 * 220 V relay	80	08	30
PISO-P8SSR8AC	8* isolated D/I + 8 * SSR /AC	80	08	30
PISO-P8SSR8DC	8* isolated D/I + 8 * SSR /DC	80	08	30
PISO-730	16*DI + 16*D/O + 16* isolated D/I + 16*isolated D/O (**Current sinking**)	80	08	40
PISO-730A	16*DI + 16*D/O + 16* isolated D/I + 16*isolated D/O (**Current sourcing**)	80	08	80
PISO-813	32 * isolated A/D	80(C280)	0A(00)	00(02)
PISO-DA2	2 * isolated D/A	80	0B	00

注：如果你的板卡版本不同，它可能有不同的 Sub ID。可是不会出现使用问题，对于不同版本的板卡我们会提供同样的功能调用。

3.2 分配 I/O 地址

即插即用 BIOS 将分配 PIO/PISO 板卡适当的 I/O 地址。假如只有一块 PIO/PISO 卡，用户能够确定这块卡为 card_0。如果有两个 PIO/PISO 卡在系统中，用户将很难找到哪一块卡是 card_0。软件驱动最多能够支持 16 块卡。所以用户可以安装 16 块 PIO/PISO 板卡在一台 PC 中。下面的方法说明查找和确定 card_0 和 card_1：

下面的方式是使用 wSlotBus 和 wSlotDevice 来简单的识别 card_0：

步骤 1: 移除 PC 中所有 PIO-D64 系列卡。

步骤 2: 安装一块 PIO-D64 系列卡到 PC 的第一个 PCI 插槽，运行 PIO_PISO.EXE。然后记录 wSlotBus1 和 wSlotDevice1 的信息。

步骤 3: 移除 PC 中所有 PIO-D64 系列卡。

步骤 4: 安装一块 PIO-D64 系列在 PC 的第二个插槽并运行 PIO_PISO.EXE。然后记录 wSlotBus2 和 wSlotDevice2 的信息。

步骤 5: 重复步骤 3 和 4 到每个 PCI 插槽并记录所有 wSlotBus 和 wSlotDevice 信息。

记录的信息也许与下表相似：

表 3-2

PC's PCI slot	wSlotBus	wSlotDevice
Slot_1	**0**	**0x07**
Slot_2	**0**	**0x08**
Slot_3	**0**	**0x09**
Slot_4	**0**	**0x0A**
PCI-BRIDGE		
Slot_5	**1**	**0x0A**
Slot_6	**1**	**0x08**
Slot_7	**1**	**0x09**
Slot_8	**1**	**0x07**

上面所记录的是在一台 PC 机上的 wSlotBus 和 wSlotDevice 信息。这些值将被映射到 PC 的物理插槽。任何 PIO/PISO 卡的映射将不能被改变。因此，下面三个步骤就能够使用这个信息确定 PIO/PISO 卡：

步骤 1: 利用表 3-2 wSlotBus 和 wSlotDevice 信息
步骤 2: 输入板卡号到函数 PIO_GetConfigAddressSpace(…)去获得板卡的详细信息，尤其是 wSlotBus 和 wSlotDevice 信息。
步骤 3: 用户可以识别一个指定的 PIO/PISO 板卡，通过步骤 1 和步骤 2 得到的 wSlotBus 和 wSlotDevice 数据来比较。

注：通常 PIO/PISO 卡安装在插槽 0 就是 card0，安装在插槽 1 就是 card1。

3.3 I/O 地址映像

通过主板 ROM BIOS 去自动分配 PIO/PISO 板卡的 I/O 地址。用户同样可以再分配 I/O 地址。强烈推荐用户不要去改变 I/O 地址，即插即用 BIOS 将会很好的去分配每个 PIO/PISO 板卡最适合的 I/O 地址。PIO-D64 系列卡的 I/O 地址见下表，

表 3-3

地址	Read/读	Write/写
Wbase+0	RESET\ 控制寄存器	相同
Wbase+2	Aux 控制寄存器	相同
Wbase+3	Aux 资料寄存器	相同
Wbase+5	INT 屏蔽控制寄存器	相同
Wbase+7	Aux 引脚状态寄存器	相同
Wbase+0x2a	INT 极性控制寄存器	相同
Wbase+0xc0	DI0~DI7	DO0~DO7
Wbase+0xc4	DI8~DI15	DO8~DO15
Wbase+0xc8	DI16~DI23	DO16~DO23
Wbase+0xcc	DI24~DI31	DO24~DO31
Wbase+0xd0	读取 U4 8254-counter0	写入 U4 8254-counter0
Wbase+0xd4	读取 U4 8254-counter1	写入 U4 8254-counter1
Wbase+0xd8	读取 U4 8254-counter2	写入 U4 8254-counter2
Wbase+0xdc	读取 U4 8254 control word	写入 U4 8254 control word
Wbase+0xe0	读取 U5 8254-counter0	写入 U5 8254-counter0
Wbase+0xe4	读取 U5 8254-counter1	写入 U5 8254-counter1
Wbase+0xe8	读取 U5 8254-counter2	写入 U5 8254-counter2
Wbase+0xec	读取 U5 8254 control word	写入 U5 8254 control word
Wbase+0xf4	读取 Card ID	

注意：有关 wBase 详情请参考章节 3.1

3.3.1 RESET\ 控制寄存器

(Read/Write): wBase+0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
保留	保留	保留	保留	保留	保留	保留	RESET\

注. 详细关于 wBase 请参考至 章节 3.1 。

当 PC 的电源第一次开启，RESET\ 信号是低电平状态。**这将禁用所有 D/I/O 操行**。用户在使用任何 D/I/O 命令前，必须将 RESET\ 信号置于高电平状态。

```
outportb (wBase,1);          /*  RESET\=High → 所有 D/I/O 启动 */
outportb (wBase,0);          /*  RESET\=Low → 所有 D/I/O 禁用 */
```

3.3.2 AUX 控制寄存器

(Read/Write): wBase+2

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Aux7	Aux6	Aux5	Aux4	Aux3	Aux2	Aux1	Aux0

注. 详细关于 wBase 请参考至 章节 3.1 。

Aux?=0→ 表明 Aux 用作 D/I

Aux?=1→ 表明 Aux 用作 D/O

当 PC 第一次上电启动时，所有 Aux 信号默认为低电平，且所有 PIO/PISO 系列板卡 I/O 口为数字量输入状态。

3.3.3 AUX 资料寄存器

(Read/Write): wBase+3

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Aux7	Aux6	Aux5	Aux4	Aux3	Aux2	Aux1	Aux0

注. 详细关于 wBase 请参考至 章节 3.1 。

当 Aux?用于 D/O 时，输出状态由寄存器控制。由于寄存器设计结构特点所至，因此这个寄存器为保留不使用。

3.3.4 INT 屏蔽控制寄存器

(Read/Write): wBase+5

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	EN3	EN2	EN1	EN0

注. 详细关于 wBase 请参考至 章节 3.1 。

EN0=0→ 禁用 INT_CHAN_0 作为中断信号 (默认)
EN0=1→ 启动 INT_CHAN_0 作为中断信号

EN1=0→ 禁用 INT_CHAN_1 作为中断信号 (默认)
EN1=1→ 启动 INT_CHAN_1 作为中断信号

EN2=0→ 禁用 INT_CHAN_2 作为中断信号 (默认)
EN2=1→ 启动 INT_CHAN_2 作为中断信号

```
outportb(wBase+5,0);          /* 禁用所有中断                */
outportb(wBase+5,1);          /* 启动中断 INT_CHAN_0         */
outportb(wBase+5,2);          /* 启动中断 INT_CHAN_1         */
outportb(wBase+5,4);          /* 启动中断 INT_CHAN_2         */
outportb(wBase+5,7);          /* 激活 3 个通道中断           */
```

参考下面 DEMO 程序:

DEMO3.C of DOS → INT_CHAN_0
DEMO4.C of DOS → INT_CHAN_1
DEMO5.C of DOS → INT_CHAN_2
DEMO6.C of DOS → INT_CHAN_1 和 INT_CHAN_2

3.3.5 Aux 状态寄存器

(Read/Write): wBase+7

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Aux7	Aux6	Aux5	Aux4	Aux3	Aux2	Aux1	Aux0

注. 详细关于 wBase 请参考至 章节 3.1 。

Aux0=INT_CHAN0, Aux1=INT_CHAN_1, Aux2=INT_CHAN_2, Aux3=INT_CHAN_3, Aux7~4=Aux-ID. Aux 0~3 作为中断源, 为分辨中断源中断服务程序必须读取该这个寄存器。
更多信息参考 章节 2.9 。

3.3.6 中断极性控制寄存器

(Read/Write): wBase+0x2a

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	INV3	INV2	INV1	INV0

注. 详细关于 wBase 请参考至 章节 3.1 。

该寄存器提供函数控制中断信号源反相或正相作用。具体应用案例如下：

INV0/1/2=0→ 选择 INT_CHAN_0/1/2 的反相信号
INV0/1/2=1→ 选择 INT_CHAN_0/1/2 的正相信号

```
outportb(wBase+0x2a,0);      /* 选择所有 3 信道的中断源信号反相输入  */
outportb(wBase+0x2a,0x0f);  /* 选择所有 3 信道中断源信号正相输入  */
outportb(wBase+0x2a,0x0e);  /* 选择反相输入 INT_CHAN_0        */
                                    /* 选择其它通道正相输入  */
outportb(wBase+0x2a,0x0c);  /* 选择反相输入 INT_CHAN_0, INT_CHAN_1   */
                                    /* 选择其它通道正相输入     */
```

更多信息参考 章节 2.9 与 DEMO6.C。

3.3.7 读/写 8254

■ **8254 控制命令**

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SC1	SC0	RL1	RL0	M2	M1	M0	BCD

注. 详细关于 wBase 请参考至 章节 3.1 。

SC1,SC0 : 00: counter0
01: counter1
10: counter2
11: 读回命令

RL1,RL0 : 00: 计数器锁存指令
01: 仅读/写计数器低位字节
10: 仅读/写计数器高位字节
11: 先读/写计数器低位字节，再读/写计数器高位字节

M2,M1,M0: 000: mode0 终端计数中断
001: mode1 一次性可编程
010: mode2 速率信号发生器
011: mode3 方波发生器
100: mode4 软件触发脉冲
101: mode5 硬件触发脉冲

BCD : 0: 二进制计数 1: BCD 码计数

3.3.8 读 Card ID 寄存器

(Read): wBase+0xf4

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	**ID3**	**ID2**	**ID1**	**ID0**

注. 详细关于 wBase 请参考至 章节 3.1 。

```
wCardID = inportb(wBase+0xF4);          /* 读取 Card ID */
```

注: 仅有 PIO-D64U (1.0 版或更新版本) 支持 Card ID 功能。

4 软件安装向导

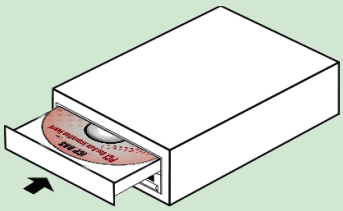
PIO-D64 系列板卡支持在 DOS、Windows 98/NT/2000、32-bit 及 64-bit Windows XP/2003/Vista/2008/7 等操作系统环境下使用。软件安装程序参考本手册中第 4.1 节 ~ 第 4.3 节，或参考快速入门指南(CD:\NAPDOS\PCI\PIO-DIO\Manual\QuickStart\)
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pio-dio/manual/quickstart/

4.1 软件驱动程序安装

1. 依照您的 PC 平台选择适合的驱动程序来安装。

■ UniDAQ SDK 驱动程序 (支持 32-bit/64-bit Windows XP/2003/Vista/7):

步骤 1: 将软件安装 PCI 光盘放入 CD-ROM 光驱中，PCI 光盘安装程序将自动启动。如 PCI 光盘安装程序无法自动启动，请以手动方式来启动: 1.进入 PCI 光盘中 NAPDOS 文件夹里。2. 双击 NAPDOS 活页夹中的 **AUTO32.EXE** 执行档。

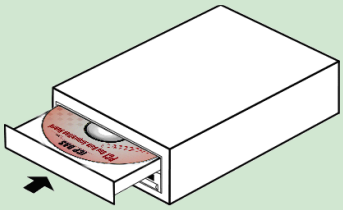


步骤 2: 单击 **“PCI Bus DAQ Card”** 项目。
步骤 3: 单击 **“UniDAQ”** 项目。
步骤 4: 单击 **“DLL for Windows 2000 and XP/2003/Vista 32-bit”** 项目。

步骤 5: 双击 **Driver** 活页夹中的 **“UniDAQ_Win_Setup_x.x.x.x._xxxx.exe”** 执行档。

■ PIO-DIO Series 驱动程序 (支持 Windows 98/NT/2K 及 32-bit Windows XP/2003/Vista/7):

步骤 1: 将软件安装 PCI 光盘放入 CD-ROM 光驱中，PCI 光盘安装程序将自动启动。如 PCI 光盘安装程序无法自动启动，请以手动方式来启动: 1.进入 PCI 光盘中 NAPDOS 文件夹里。2. 双击 NAPDOS 活页夹中的 AUTO32.EXE 执行档。



步骤 2: 单击 **“PCI Bus DAQ Card”** 项目。

步骤 3: 单击 **“PIO-DIO”** 项目。

步骤 4: 单击 **“DLL and OCX for Windows 98/NT/2K/XP/2003”** 项目。

步骤 5: 在 **Driver** 活页夹中，双击 **“PIO_DIO_Win_vxxx.exe”** 执行档来安装。

2. 双击安装执行档后，将开始安装驱动程序并复制相关档 (”.DLL”, ”.SYS”, “.Vxd”) 到指定的目录下及注册驱动程序至您的 PC 上，目录路径如下所示:

■ **64-bit Windows XP/2003/Vista/7:**

UniDAQ.DLL 檔将被复制到 C:\WINNT\SYSTEM32 活页夹下

NAPWNT.SYS 及 UniDAQ.SYS 檔将被复制到 C:\WINNT\SYSTEM32\DRIVERS 活页夹下

更多更详细的 UniDAQ.DLL 函式信息，请参考到 UniDAQ SDK 用户手册。 (CD:\NAPDOS\PCI\UniDAQ\Manual\).

http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/unidaq/maunal/

■ **Windows NT/2K 和 32-bit Windows XP/2003/Vista/7:**

PIODIO.DLL 檔将被复制到 C:\WINNT\SYSTEM32 活页夹下

NAPWNT.SYS 及 PIO.SYS 檔将被复制到

C:\WINNT\SYSTEM32\DRIVERS 活页夹下

■ **Windows 95/98/ME:**

PIODIO.DLL 及 PIODIO.Vxd 檔将被复制到

C:\Windows\SYSTEM 活页夹下

更多更详细的 PIODIO.DLL 函式信息，请参考到 PIO-DIO 系列 DLL 软件手册。

(CD:\NAPDOS\PCI\PIO-DIO\Manual\)**.**

http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pio-dio/manual/

4.2 PnP 驱动程序安装

以上软件驱动程序安装完成后，请关闭您 PC 的电源，并安装 PIO-D48 系列板卡到 PC 的插槽上。板卡安装完成后，并启动 PC 电源，当进入 Windows 系统后，Plug&Play 驱动程序会自动执行，板卡即可使用。如 Plug&Play 发生问题造成无法自动执行，请以手动方式来安装，请参考到 PnPInstall.pdf 。

4.3 确认板卡安装成功

请到装置管理员中来确认您的 PIO-D64 系列板卡已正确的安装到 PC 中，开启装置管理员步骤如下:

步骤 1: 选择 「开始」→ 「设定(S)」→「控制台(C)」后，双击控制台窗口中的「系统」。

步骤 2: 在系统内容窗口中，选择「硬件」卷标，并单击「装置管理员(D)」按钮来进入装置管理员中。

步骤 3: 检查 PIO-D64 系列板卡是否正确安装，如已安装完成，装置管理员中将显示 PIO-D64 板卡名称于 DAQCard 项目下，如下图所示:

5 示例程序

5.1 Windows DEMO 程序

如果 DLL 驱动没有正确安装那么所有 DEMO 程序将不能正常工作。 在 DLL 驱动安装过程的时候，安装程序将注册适当的内核驱动到操作系统中，并且拷贝 DLL 驱动和 DEMO 程序到适当的位置，你可以选择(Win98/Me/NT/2000 and 32-bit Win XP/2003/Visa/7)驱动软件包。一次完整的驱动安装，下列出现相关的 DEMO 程序、库文件、声明的头档在不同的运行环境：

取得 Windows 示例程序位置:

CD:\NAPDOS\PCI\PIO-DIO\DLL_OCX\Demo\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pio-dio/dll_ocx/demo/

- BCB 4 → for Borland C++ Builder 4
  PIODIO.H → Header files
  PIODIO.LIB → Linkage library for BCB
- Delphi4 → for Delphi 4
  PIODIO.PAS → Declaration files
- VB6 → for Visual Basic 6
  PIODIO.BAS → Declaration files
- VC6 → for Visual C++ 6
  PIODIO.H → Header files
  PIODIO.LIB → Linkage library for VC only
- VB.NET2005 → for VB.NET2005
  PIODIO.vb → Visual Basic Source files
- CSharp2005 → for C#.NET2005
  PIODIO.cs → Visual C# Source files

DEMO 程序清单如下:

- DIO Demo
- Int Demo
- IntAPC Demo
- Counter Dome
- 32bitCounter Demo

5.2 DOS DEMO 程序

DOS 软件和所有 DEMO 在 CD 中可以找到：

CD:\NAPDOS\PCI\PIO-DIO\dos\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pio-dio/dos/

- TC*.* → Turbo C 2.xx 或更新版本
  TC\LIB*.* → TC 库文件
  TC\DEMO*.* → TC Demo 程序
  TC\DIAG*.* → TC 诊断程序

  TC\LIB\PIO.H → TC 声明檔
  TC\\LIB\TCPIO_L.LIB → TC 大型模块库文件
  TC\\LIB\TCPIO_H.LIB → TC 巨型模块库文件

- MSC*.* → for MSC 5.xx 或更新版本
  MSC\LIB\PIO.H → MSC 声明檔
  MSC\\LIB\MSCPIO_L.LIB → MSC 大型模块库文件
  MSC\\LIB\MSCPIO_H.LIB → MSC 巨型模块库文件

- BC*.* → for BC 3.xx 或更新版本
  BC\LIB\PIO.H → BC 声明檔
  BC\\LIB\BCPIO_L.LIB → BC 大型模块库文件
  BC\\LIB\BCPIO_H.LIB → BC 巨型模块库文件

DEMO 程序清单如下:

DEMO1: D/O demo
DEMO2: D/I/O demo
DEMO3: 使用外部中断(external int)来量测脉冲宽度 (high level)
DEMO4: 使用 EVTIRQ 计数事件
DEMO5: 使用 TMRIRQ 来产生 0.5Hz 方波
DEMO6: 使用 TMRIRQ 来产生 0.5Hz 方波 及 EVTIRQ 计数

附录

A1. DOS LIB 函式

A1-1. ErrorCode 和 ErrorString Code 描述表

表 A.1

Error Code	Error ID	Error String
0	NoError	OK (No error)
1	Driver HandleError	Error opening the device driver
2	DriverCallError	An error occurred while calling the driver functions
3	FindBoardError	Can't find the board on the system
4	TimeOut	Timeout
5	ExeedBoardNumber	Invalid board number (Valid range: 0 to TotalBoards -1)
6	NotFoundBoard	Can't detect the board on the system

A1-2. PIO_DriverInit

■ **描述:**

这个函数能检测所有系统中 PIO/PISO 板卡。它是基于 PCI 即插即用装置上。它将找到所有安装在系统中的 PIO/PISO 板卡和保存所有它们的资源在库中。

■ **语法:**

WORD **PIO_DriverInit**(WORD ***wBoards**, WORD **wSubVendorID**,
WORD **wSubDeviceID**, WORD **wSubAuxID**)

■ **参数:**

WBoards	[Output]	PC 中板卡号
wSubVendorID	[Input]	板卡的 SubVendor ID
wSubDeviceID	[Input]	板卡的 SubDevice ID
wSubAuxID	[Input]	板卡的 SubAux

■ **返回值:**

请参考至 "表 A.1"

A1-3. PIO_GetConfigAddressSpace

■ **描述:**

用户能够使用这个函数去得到安装在系统中所有 PIO/PISO 板卡资源信息。那么应用程序就能够很方便的去调用 PIO/PISO 函数.

■ **语法:**

WORD **PIO_GetConfigAddressSpace**(**wBoardNo**,***wBase**,***wIrq**, **wSubVendor**, ***wSubDevice**, ***wSubAux**, ***wSlotBus**, ***wSlotDevice**)

■ **参数:**

wBoardNo	[Input]	板卡号
wBase	[Output]	板卡的基地址
wIrq	[Output]	板卡使用的 IRQ 号
wSubVendor	[Output]	Sub Vendor ID
wSubDevice	[Output]	Sub Device ID
wSubAux	[Output]	Sub Aux ID
wSlotBus	[Output]	Slot Bus number (插槽号)
wSlotDevice	[Output]	Slot Device ID (插槽设备 ID)

■ **返回值:**

请参考至 "表 A.1"

A1-4. PIO_GetDriverVersion

■ **描述:**

这个函数将得到 PIODIO 驱动版本号.

■ **语法:**

WORD **PIO_GetDriverVersion**(WORD ***wDriverVersion**)

■ **参数:**

wDriverVersion	[Output]	wDriverVersion 地址

■ **返回值:**

请参考至 "表 A.1"

A1-5. ShowPIOPISO

- **描述:**
  这个函数将显示一个专用的 Sub_ID 文本字符串.这个文本字符串同 PIO.H 中定义的一样.

- **语法:**
  WORD **ShowPIOPISO**(**wSubVendor**, **wSubDevice**, **wSubAux**)

- **参数:**

wSubVendor	[Input]	板卡的 SubVendor ID
wSubDevice	[Input]	板卡的 SubDevice ID
wSubAux	[Input]	板卡的 SubAux ID

- **返回值:**
  请参考至 "表 A.1"

A1-6. PIO_DriverClose

- **描述:**
  这个函数用来关闭 PIODIO 设备并释放计算机驱动的资源。用户的应用程序关闭之前需要调用这个函数一次.

- **语法:**
  WORD **PIO_DriverClose**()

- **参数:**
  无

- **返回值:**
  请参考至 "表 A.1"