ACCES I/O PRODUCTS INC
10623 Roselle Street, San Diego, CA 92121
TEL (858)550-9559 FAX (858)550-7322

MODEL PIOD24

USER MANUAL

FILE: MPIOD24.B1a

Notice

The information in this document is provided for reference only. ACCES does not assume any liability arising out of the application or use of the information or products described herein. This document may contain or reference information and products protected by copyrights or patents and does not convey any license under the patent rights of ACCES, nor the rights of others.

IBM PC, PC/XT, and PC/AT are registered trademarks of the International Business Machines Corporation.

Printed in USA.

U.S. Patent No.s 4,603,320 and 4,972,470.

Warranty

Prior to shipment, ACCES equipment is thoroughly inspected and tested to applicable specifications. However, should equipment failure occur, ACCES assures its customers that prompt service and support will be available. All equipment originally manufactured by ACCES which is found to be defective will be repaired or replaced subject to the following considerations.

Terms and Conditions

If a unit is suspected of failure, contact ACCES' Customer Service department. Be prepared to give the unit model number, serial number, and a description of the failure symptom(s). We may suggest some simple tests to confirm the failure. We will assign a Return Material Authorization (RMA) number which must appear on the outer label of the return package. All units/components should be properly packed for handling and returned with freight prepaid to the ACCES designated Service Center, and will be returned to the customer's/user's site freight prepaid and invoiced.

Coverage

First Three Years: Returned unit/part will be repaired and/or replaced at ACCES option with no charge for labor or parts not excluded by warranty. Warranty commences with equipment shipment.

Following Years: Throughout your equipment's lifetime, ACCES stands ready to provide on-site or in-plant service at reasonable rates similar to those of other manufacturers in the industry.

Equipment Not Manufactured by ACCES

Equipment provided but not manufactured by ACCES is warranted and will be repaired according to the terms and conditions of the respective equipment manufacturer's warranty.

General

Under this Warranty, liability of ACCES is limited to replacing, repairing or issuing credit (at ACCES discretion) for any products which are proved to be defective during the warranty period. In no case is ACCES liable for consequential or special damage arriving from use or misuse of our product. The customer is responsible for all charges caused by modifications or additions to ACCES equipment not approved in writing by ACCES or, if in ACCES opinion the equipment has been subjected to abnormal use. "Abnormal use" for purposes of this warranty is defined as any use to which the equipment is exposed other than that use specified or intended as evidenced by purchase or sales representation. Other than the above, no other warranty, expressed or implied, shall apply to any and all such equipment furnished or sold by ACCES.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

Features

- Type II PCMCIA Card
- Plug-N-Play Windows 95/98 Installation
- 24 Bit Digital I/O port
- Four and eight-bit groups independently selectable for digital I/O
- Mode 1 and 2 Bi-directional and strobed data transfers supported
- Three 16-bit 10MHz counters
- Software setup including FINDBASE and TSR setup program
- 37 pin D-sub Male Connector via cable adaptors
- Supports external counter clocks, gates and outputs, and internal control.

Introducing PCMCIA

PCMCIA started as a way to add extra memory to laptop and portable computers. As the market grew, the absence of a standard expansion slot was recognized and the second release added hardware support for I/O devices. The PIOD24 is one of a family of PCMCIA devices offered that extend the ports and functionality of PC systems. This expansion port may be referred on your computer system as either a PCMCIA or PC CARD port.

Software Compatibility

Drivers and example programs are supplied with the PIOD24 in a variety of programming languages and operating systems; including DOS, 16-bit and 32-bit Windows (Windows 3.x vs Windows 95 and Windows NT).

Many packaged application programs support similar cards, but to use the full functionality requires a customized driver. Please call ACCES to acquire the appropriate driver or DLL for your program.

Card and Socket Services

PCMCIA Card and Socket services must be loaded on the host computer system before using the PIOD24 card and is typically supplied by the PCMCIA slot provider (the laptop or adaptor manufacturer) to provide the software interface to the card slot drive. In addition, our TSR memory-resident program, described below, is required to configure the card once it is recognized by the card and socket services.

The Card and Socket services handle the PCMCIA slot software interface with your operating system when a new card is inserted (or removed) or when power is applied (or upon shut-down). Once the PCMCIA Card and Socket Services recognizes the PIOD24 card it will then appear to your application software like a card on the internal ISA bus.

Calibration and Service

No calibration is required for PCMCIA devices and especially for the all-digital PIOD24. The case may not be opened and there are no parts inside which you can service. There are no socketed components. Opening the PIOD24 case will void your warranty.

If your unit requires service, please return it to ACCES. Please call ACCES for an RMA number before returning the card for service, even if under warranty.

Utility Software

We include programs on CD to support the card and to help you develop your application program(s). The card supports I/O bus addresses from 000 to 3FF (hex) and a program called FINDBASE helps you find an open base address to use with the card.

The TSR client program has two required command-line parameters (entered in hexadecimal format); the base I/O address and the interrupt request (IRQ) number. Optionally, a "verbose" switch will turn on a detailed report of the communication between the TSR and the PCMCIA card for diagnostic troubleshooting. A value of zero for the IRQ number will configure the card not to use any interrupts.

Syntax:	PIOD24 [Base Address in Hex] [IRQ number in Hex] (verbose)
For example:	PIOD24 300 0A verbose

Would configure the PIOD24 card at a base address of 300 (hex) and IRQ at 10 (decimal). If invalid values are used with the TSR program then the following error message is displayed:

The command line parameters were incorrect.
syntax: piod24 <port address> <IRQ>
example: piod24 120 5
example: piod24 300 0A

If no values are used with the TSR program, then the program reads the CIS information from the PIOD24 and attempts to automatically configure address and IRQ.

Specifications

Features

- 24 channels of unbuffered digital input/output.
- Four and Eight bit groups independently selectable for I/O.
- Mode 1 and 2 Bi-directional and strobed I/O data transfers supported.
- Interrupt and Interrupt-disable capability.
- External and internal interrupt and counter source capacity.
- Three 16-bit counter/timers.

Digital Inputs/Outputs

- Logic High: 2.0 to 7.0 VDC.
- Logic Low: -0.3 to 0.8 VDC.
- Input Load (Hi): +3.2 microamperes.
- Input Load (Lo): -250 microamperes.
- Max Drive: 5 LSTTL Loads.

Counters

- Number: Three.
- Type: 16 bit, 65535 counts.
- Frequency: 10 MHz Max, Internal or external source.
- Output High: 2.4 V Min @ -250 mA.
- Output Low: 2.0 V Min, 7.3 Max.

Interrupt

- Level: 2-6, 7, 10-15 set by CFG configuration file.
- Enable: Programmable.
- Trigger: External or Internal.

Environmental

- Operating Temperature Range: 0 °C. to +60 °C.
- Storage Temperature Range: -50 °C. to +120 °C.
- Humidity: 5% to 95%, non-condensing.
- Power Required: 5 VDC, 47 mA Typical, 65 mA Max
- Size Type II PCMCIA card slot.

Regulatory Compliance

- FCC Part 15, Level A: Designed to meet.
- CE Certification not completed at print time.
- Please check with us for certification status.

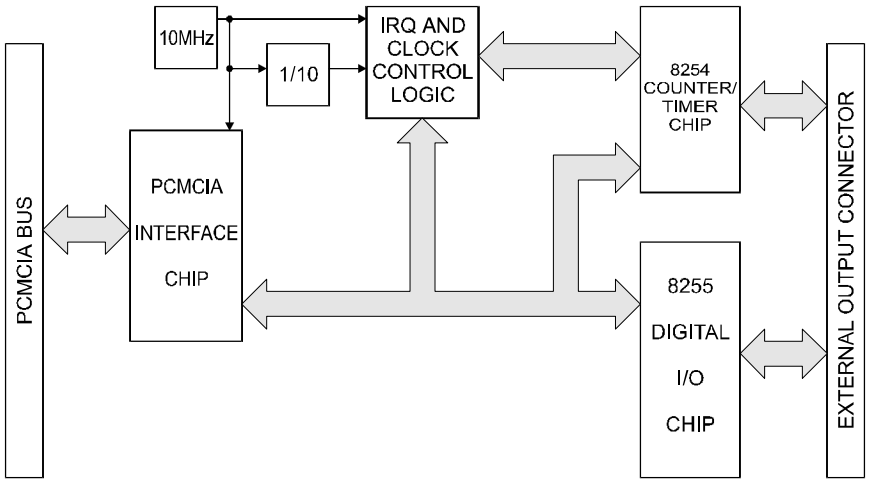


Figure 1-1: PIOD24 Block Diagram

Chapter 2: Installation

The software provided with this card is contained on either one CD or multiple diskettes and must be installed onto your hard disk prior to use. To do this, perform the following steps as appropriate for your software format and operating system. Substitute the appropriate drive letter for your CD-ROM or disk drive where you see d: or a: respectively in the examples below.

CD Installation

DOS/WIN3.x

1. Place the CD into your CD-ROM drive.
2. Type ? ? ? to change the active drive to the CD-ROM drive.
3. Type ? ? ? ? ? ? ? ? to run the install program.
4. Follow the on-screen prompts to install the software for this card.

WIN95/98/NT

a. Place the CD into your CD-ROM drive.
b. The CD should automatically run the install program after 30 seconds. If the install program does not run, click START | RUN and type d:install, click OK or press ? .
c. Follow the on-screen prompts to install the software for this card.

3.5-Inch Diskette Installation

As with any software package, you should make backup copies for everyday use and store your original master diskettes in a safe location. The easiest way to make a backup copy is to use the DOS DISKCOPY utility.

In a single-drive system, the command is:

? ? ? ? ? ? ? ? ? ? ? ? ? ? ?

You will need to swap disks as requested by the system.
In a two-disk system, the command is:

? ? ? ? ? ? ? ? ? ? ? ? ? ? ?

This will copy the contents of the master disk in drive A to the backup disk in drive B.

To copy the files on the master diskette to your hard disk, perform the following steps.

a. Place the master diskette into a floppy drive.

b. Change the active drive to the drive that has the diskette installed. For example, if the diskette is in drive A, type ? ? ? .

c. Type ? ? ? ? ? ? ? ? and follow the on-screen prompts.

Directories Created on the Hard Disk

The installation process will create several directories on your hard disk. If you accept the installation defaults, the following structure will exist.

[CARDNAME]

Root or base directory containing the SETUP.EXE setup program used to help you configure jumpers and calibrate the card.

DOS\PSAMPLES: A subdirectory of [CARDNAME] that contains Pascal samples.

DOS\CSAMPLES: A subdirectory of [CARDNAME] that contains "C" samples.

Win32\language: Subdirectories containing samples for Win95/98 and NT.

WinRisc.exe

A Windows dumb-terminal type communication program designed for RS422/485 operation. Used primarily with Remote Data Acquisition Pods and our RS422/485 serial communication product line. Can be used to say hello to an installed modem.

ACCES32

This directory contains the Windows 95/98/NT driver used to provide access to the hardware registers when writing 32-bit Windows software. Several samples are provided in a variety of languages to demonstrate how to use this driver. The DLL provides four functions (InPortB, OutPortB, InPort, and OutPort) to access the hardware.

This directory also contains the device driver for Windows NT, ACCESNT.SYS. This device driver provides register-level hardware access in Windows NT. Two methods of using the driver are available, through ACCES32.DLL (recommended) and through the DeviceIOControl handles provided by ACCESNT.SYS (slightly faster).

SAMPLES

Samples for using ACCES32.DLL are provided in this directory. Using this DLL not only makes the hardware programming easier (MUCH easier), but also one source file can be used for both Windows 95/98 and WindowsNT. One executable can run under both operating systems and still have full access to the hardware registers. The DLL is used exactly like any other DLL, so it is compatible with any language capable of using 32-bit DLLs. Consult the manuals provided with your language's compiler for information on using DLLs in your specific environment.

VBACCES

This directory contains sixteen-bit DLL drivers for use with VisualBASIC 3.0 and Windows 3.1 only. These drivers provide four functions, similar to the ACCES32.DLL. However, this DLL is only compatible with 16-bit executables. Migration from 16-bit to 32-bit is simplified because of the similarity between VBACCES and ACCES32.

PCI

This directory contains PCI-bus specific programs and information. If you are not using a PCI card, this directory will not be installed.

SOURCE

A utility program is provided with source code you can use to determine allocated resources at run-time from your own programs in DOS.

PCIFind.exe

A utility for DOS and Windows to determine what base addresses and IRQs are allocated to installed PCI cards. This program runs two versions, depending on the operating system. Windows 95/98/NT displays a GUI interface, and modifies the registry. When run from DOS or Windows3.x, a text interface is used. For information about the format of the registry key, consult the card-specific samples provided with the hardware. In Windows NT, NTioPCI.SYS runs each time the computer is booted, thereby refreshing the registry as PCI hardware is added or removed. In Windows 95/98/NT PCIFind.EXE places itself in the boot-sequence of the OS to refresh the registry on each power-up.

This program also provides some COM configuration when used with PCI COM ports. Specifically, it will configure compatible COM cards for IRQ sharing and multiple port issues.

WIN32IRQ

This directory provides a generic interface for IRQ handling in Windows 95/98/NT. Source code is provided for the driver, greatly simplifying the creation of custom drivers for specific needs. Samples are provided to demonstrate the use of the generic driver. Note that the use of IRQs in near-real-time data acquisition programs requires multi-threaded application programming techniques and must be considered an intermediate to advanced programming topic. Delphi, C++ Builder, and Visual C++ samples are provided.

Findbase.exe

DOS utility to determine an available base address for ISA bus , non-Plug-n-Play cards. Run this program once, before the hardware is installed in the computer, to determine an available address to give the card. Once the address has been determined, run the setup program provided with the hardware to see instructions on setting the address switch and various option selections.

Poly.exe

A generic utility to convert a table of data into an nth order polynomial. Useful for calculating linearization polynomial coefficients for thermocouples and other non-linear sensors.

Risc.bat

A batch file demonstrating the command line parameters of RISCTerm.exe.

RISCTerm.exe

A dumb-terminal type communication program designed for RS422/485 operation. Used primarily with Remote Data Acquisition Pods and our RS422/485 serial communication product line. Can be used to say hello to an installed modem. RISCTerm stands for Really Incredibly Simple Communications TERMinal.

Installing the Card

The PIOD24 card can be installed in any PCMCIA Type II card slot. There are no switches or jumpers to set. Everything on the PIOD24 card is programmable including address and interrupt level. If you have two PCMCIA slots you can plug two PIOD24 in the same Windows based computer.

DOS or Windows 3.x Installation

You must have installed the Card & Socket Services (CSS) software that was provided with your PCMCIA-capable computer. For complete installation instructions refer to "readme.txt" in your PIOD24 software directory.

DOS 'Plug N Play' or Superclient Installation

The SuperClient uses the PIOD24 internal CIS (configuration information) to configure the card and prepare system for use. Consult the SuperClient documentation to determine the installation procedure.

Windows 95 Installation

Windows95 will automatically recognize the installation of the PCMCIA card and provide a list of options to install the card. Select the option "Driver from disk provided by hardware manufacturer" and select the CD provided which will install support for the card. Depending on options selected in the Windows95 PCMCIA drivers, you may hear a "rising" two-note tone upon successful installation and a new PCMCIA icon in the taskbar tray.

Windows NT 4.0 Installation

Windows NT 4.0 is not a Plug-and-Play operating system, so getting the card to work properly requires a few extra steps. After physically installing the PIOD24 card into the slot, run the "Setup.exe" program included in the software package. This program will create an entry in the PCMCIA Database, allowing Windows NT 4.0 to recognize the PIOD24. After running "Setup.exe," the system must be rebooted for the changes to take effect. The device should then show up in the "PC Card (PCMCIA)" Control Panel where its resource assignments can be found.

Although manual modification of the Registry should not be necessary, the steps necessary to duplicate the actions performed by "Setup.exe" are documented here for informational purposes. After starting Regedit, navigate to the following key:

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Pcmcia\Database

Add a new subkey named "ACCES I/O Products," then add a new subkey to the "ACCES I/O Products" key you just created named "PIOD24." In the "PIOD24" key, add a new string with the name "Driver" and the value "piod24." This completes the modifications necessary for Windows NT 4.0 to recognize the PIOD24 card. The system must then be rebooted for the changes to take effect.

Caution

The PIOD24 must be oriented with the label side up, or "side A" in PCMCIA terms. The standard PCMCIA case is "keyed" to prevent improper insertion but may be broken off if excessive force is used during installation.

Do Not Force the PIOD24 Card into the Slot.

Install the card with the label facing "up" or towards side "A" which is usually towards the keyboard, but you should check with your systems owners manual. When correctly oriented, the PIOD24 card should seat in the slot with little pressure. Force the PIOD24 card in upside down and you will probably burn out components as well as destroy the key slot.

Once the card is inserted, you may then run the SAMPLE1 software to confirm operation of your PIOD24 card.

Chapter 3: Cable Connections

The PIOD24 requires our CAB-PIOD24 cable to provide the transition between the microminiature 32 pin PCMCIA connector and a standard 37-pin D subminiature connector to interface to your project. The 37-pin connector is equipped with 4-40 threaded standoffs (female screw lock) to provide strain relief. Pins 34-37 of the cable are not connected.

	PIOD24 Connector	**CAB-PIOD24 Cable**
Pin	Description	Pin Standard DB-37 Connector
1.	Digital Port A Bit 0	1
2.	Digital Port A Bit 1	2
3.	Digital Port A Bit 2	3
4.	Digital Port A Bit 3	4
5.	Digital Port A Bit 4	5
6.	Digital Port A Bit 5	6
7.	Digital Port A Bit 6	7
8.	Digital Port A Bit 7	8
9.	Digital Port B Bit 0	9
10.	Digital Port B Bit 1	10
11.	Digital Port B Bit 2	11
12.	Digital Port B Bit 3	12
13.	Digital Port B Bit 4	13
14.	Digital Port B Bit 5	14
15.	Digital Port B Bit 6	15
16.	Digital Port B Bit 7	16
17.	Digital Port C Bit 0	17
18.	Digital Port C Bit 1	18
19.	Digital Port C Bit 2	19
20.	Digital Port C Bit 3	20
21.	Digital Port C Bit 4	21
22.	Digital Port C Bit 5	22
23.	Digital Port C Bit 6	23
24.	Digital Port C Bit 7	24
25.	Ground (Tied to pin 33)	25
26.	Counter 0 Output	26
27.	Counter 0 Gate Input	27
28.	Counter 0 External Clock Input	28
29.	Counter 2 Output	29
30.	Counter 2 Gate Input	30
31.	Counter 2 External Clock Input	31
32.	External Interrupt	32
	Ground (Tied to pin 25)	33
	No Connections	34-37

Chapter 4: Address Selection

The PIOD24 base address can be selected anywhere within an I/O address range 000-3FF hex, providing that the addresses do not overlap with other functions. If in doubt, refer to the table below for a list of standard address assignments or use the base address locator program FINDBASE provided on CD will assist you to avoid an address conflict. (The primary and secondary binary synchronous communication ports are supported by the Operating System.)

Hex Range	Usage
000-00F	DMA Chip 8237A-5
020-021	Interrupt 8259A
040-043	Timer 8253-5
060-063	PPI 8255A-5
080-083	DMA Page Register
0AX	NMI Mask Register
0CX	Reserved
0EX	Reserved
100-1FF	Not usable
200-20F	Game Control
210-217	Expansion Unit
220-24F	Reserved
278-27F	Reserved
2F0-2F7	Reserved
2F8-2FF	Asynchronous Comm'n (secondary)
300-31F	Prototype Card
320-32F	Fixed Disk
378-37F	Printer
380-38C**	SDLC Communications
380-389**	Binary Synchron's Comm'n (secondary)
3A0-3A9	Binary Synchron's Comm'n (primary)
3B0-3BF	IBM Monochrome Display/Printer
3C0-3CF	Reserved
3D0-3DF	Color/Graphics
3E0-3E7	Reserved
3F0-3F7	Diskette
3F8-3FF	Asynchronous Comm'n (primary)

** These options cannot be used together-addresses overlap

Table 4-1: Standard Address Assignments for PC and PC/XT Computers

Chapter 5: Software

There are sample programs provided with the PIOD24 Digital I/O Card in C, Pascal, QuickBASIC, and several Windows languages. DOS samples are located in the DOS directory and Windows samples are located in the WIN32 directory.

Chapter 6: Programming

Developing Your Application Software

Once the Card & Socket Services (CSS) and Client Drivers are installed the PIOD24 card may be controlled by writing to and reading from the control registers.

Port Addresses and Functions

All I/O accesses may be performed as bytes. The registers, in eight-bit format, are presented here:

Address	Write Operation	Read Operation
Base Address	82C55 Port A Set	Read back data
Base Address +1	82C55 Port B Set	Read back data
Base Address +2	82C55 Port C Set	Read back data
Base Address +3	82C55 Control Register	Write Only
Base Address +4	82C54 Counter 0 Load	Counter 0 Read back
Base Address +5	82C54 Counter 1 Load	Counter 1 Read back
Base Address +6	82C54 Counter 2 Load	Counter 2 Read back
Base Address +7	82C54 Counter Control	Write Only
Base Address +8	Interrupt & Clock Source	Write Only

Table 6-1: PIOD24 Register Address Selection

Digital I/O Port

The PIOD24 card supports operating mode 0 of the 82C55 PPI. However, it can be factory modified to accommodate either mode 1 or 2. Before attempting to use either of those modes, check to assure that the card was ordered with the appropriate modification included.

Mode 0 is the most frequently used mode of operation wherein:

a. There are two 8-bit ports (A and B) and two 4-bit ports (C Hi and C Lo).
b. Any port can be configured as an input or an output.
c. Outputs are latched.
d. Inputs are not latched.

Mode 1 is used for strobed input/output. In this mode Port A and Port B use the lines of Port C to generate and accept control signals associated with data transfer.

Mode 2 is used for bi-directional eight-bit bus applications.

The control register at base address +3 is a write-only 8-bit register. It is used to set the mode and direction of the ports. At Power-Up or Reset, all I/O lines are set as inputs. The PPI should be configured during initializing by writing to the control register even if the ports are only going to be used as inputs.

Bit assignments in the control register are as follows:

Data Bit	Option Selected		
D0	Port C (C0-C3)	1 = Input	0 = Output
D1	Port B	1 = Input	0 = Output
D2	Mode Selection	1 = Mode 1	0 = Mode 0
D3	Port C (C4-C7)	1 = Input	0 = Output
D4	Port A	1 = Input	0 = Output
D5,D6	Mode Selection	00 = Mode 0	
		01 = Mode 1	
		1X = Mode 2	
D7		Mode Set Flag 1 = Active	

Table 6-2: 8255 Control Register Bit Assignments

Note

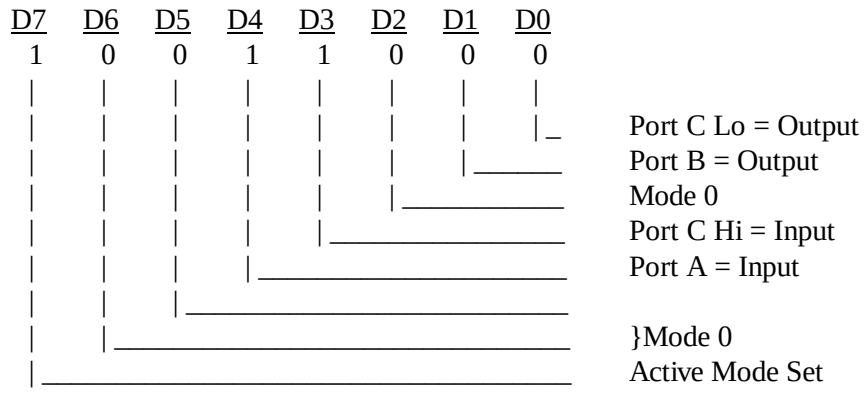
In Mode 0, do not use the control register byte for the individual bit control feature. The hardware uses the I/O bits to control buffer direction on this card. The control register should only be used for setting up input and output of the ports and enabling the buffer.

Programming Example

The following example in BASIC is provided as a guide to assist you in developing your working software. In this example, the card address is 2D0 hex, operation is in Mode 0 and the I/O lines are to be set up as follows:

Port A.....Input
Port B.....Output
Port C Hi.....Input
Port C Lo.....Output

Configure bits of the control register as follows:



This corresponds to 98 hex. If the card address is 2D0 hex, use the BASIC OUT command to write to the control register as follows:

```
BASE=0x2C0;
OUTPORTB (BASEA+3,0x98);
```

To read the inputs at Port A and the upper nybble of Port C:

```
X=INPORTB(BASE);                    //Read Port A
Y=INPORTB(BASE+2) & (0xFO);         //Read Port C Hi
```

To set outputs high (1) at Port B and the lower nybble of Port C:

```
OUTPORTB (BASE+1,0xFF);             //Turn on all Port B bits
OUTPORTB (BASE+2, 0xF);             //Turn on all bits of Port C lower nybble
```

Programmable Interval Timer

The PIOD24 contains a type 82C54 programmable counter/timer which allows you to implement such functions as a Real-Time Clock, Event Counter, Digital One-Shot, Programmable Rate Generator, Square-Wave Generator, Binary Rate Multiplier, Complex Wave Generator, and/or a Motor Controller. The 82C54 is a flexible but powerful device that consists of three independent, 16-bit, presettable, down counters. Each counter can be programmed to any count as low as 1 or 2, and up to 65,535 in binary format, depending on the mode chosen.

Operational Modes

The 82C54 modes of operation are described in the following paragraphs to familiarize you with the versatility and power of this device. For those interested in more detailed information, a full description of the 82C54 programmable interval timer can be found in the Intel (or equivalent manufacturers) data sheets. The following conventions apply for use in describing operation of the 82C54:

Clock:	A positive pulse into the counter's clock input.
Trigger:	A rising edge input to the counter's gate input.
Counter Loading:	Programming of a binary count into the counter.

Mode 0: Pulse on Terminal Count

After the counter is loaded, the output is set low and will remain low until the counter decrements to zero. The output then goes high and remains high until a new count is loaded into the counter. A trigger enables the counter to start decrementing.

Mode 1: Retriggerable One-shot

The output goes low on the clock pulse following a trigger to begin the one-shot pulse and goes high when the counter reaches zero. Additional triggers result in reloading the count and starting the cycle over. If a trigger occurs before the counter decrements to zero, a new count is loaded. Thus, this forms a re-triggerable one-shot. In mode 1, a low output pulse is provided with a period equal to the counter count-down time.

Mode 2: Rate Generator

This mode provides a divide-by-N capability where N is the count loaded into the counter. When triggered, the counter output goes low for one clock period after N counts, reloads the initial count, and the cycle starts over. This mode is periodic, the same sequence is repeated indefinitely until the gate input is brought low. This mode also works well as an alternative to mode 0 for event counting.

Mode 3: Square Wave Generator

This mode operates periodically like mode 2. The output is high for half of the count and low for the other half. If the count is even, then the output is a symmetrical square wave. If the count is odd, then the output is high for (N+1)/2 counts and low for (N-1)/2 counts. Periodic triggering or frequency synthesis are two possible applications for this mode. Note that in this mode, to achieve the square wave, the counter decrements by two for the total loaded count, then reloads and decrements by two for the second part of the wave form.

Mode 4: Software Triggered Strobe

This mode sets the output high and, when the count is loaded, the counter begins to count down. When the counter reaches zero, the output will go low for one input period. The counter must be reloaded to repeat the cycle. A low gate input will inhibit the counter.

Mode 5: Hardware Triggered Strobe

In this mode, the counter will start counting after the rising edge of the trigger input and will go low for one clock period when the terminal count is reached. The counter is retriggerable. The output will not go low until the full count after the rising edge of the trigger.

Counter/Timer Registers

Base + 4 Write/Read: Counter#0

When writing, this register is used to load a counter value into the counter. The transfer is either a single or double byte transfer, depending on the control byte written to the counter control register at Base Address + 7. If a double byte transfer is used, then the least-significant byte of the 16 bit value is written first, followed by the most significant byte. When reading, the current count of the counter is read. The type of transfer is also set by the control byte.

Base + 5 Write/Read: Counter #1

See description for Base + 4 Write/Read.

Base + 6 Write/Read: Counter#2

See description for Base + 4 Write/Read.

Base + 7 Write: Counter Control Register

The control byte specifies the counter to be programmed, the counter mode, the type of read/write operation, and the modulus. The control byte format is as follows:

Counter Control Byte

B7	B6	B5	B4	B3	B2	B1	B0
SC1	SC0	RW1	RW0	M2	M1	M0	BCD

SC0-SC1: These bits select the counter modified by the counter control byte.

SC1	SC0	Function
0	0	Program Counter 0
0	1	Program Counter 1
1	0	Program Counter 2
1	1	Read Back Command

RW0-RW1: These bits select the read/write mode of the selected counter.

RW1	RW0	Counter Read/Write Function
0	0	Counter Latch Command
0	1	Read/Write LS Byte
1	0	Read/Write MS Byte
1	1	Read/Write LS Byte, then MS Byte

M0-M2: These bits set the operational mode of the selected counter.

Mode	M2	M1	M0
0	0	0	0
1	0	0	1
2	X	1	0
3	X	1	1
4	1	0	0
5	1	0	1

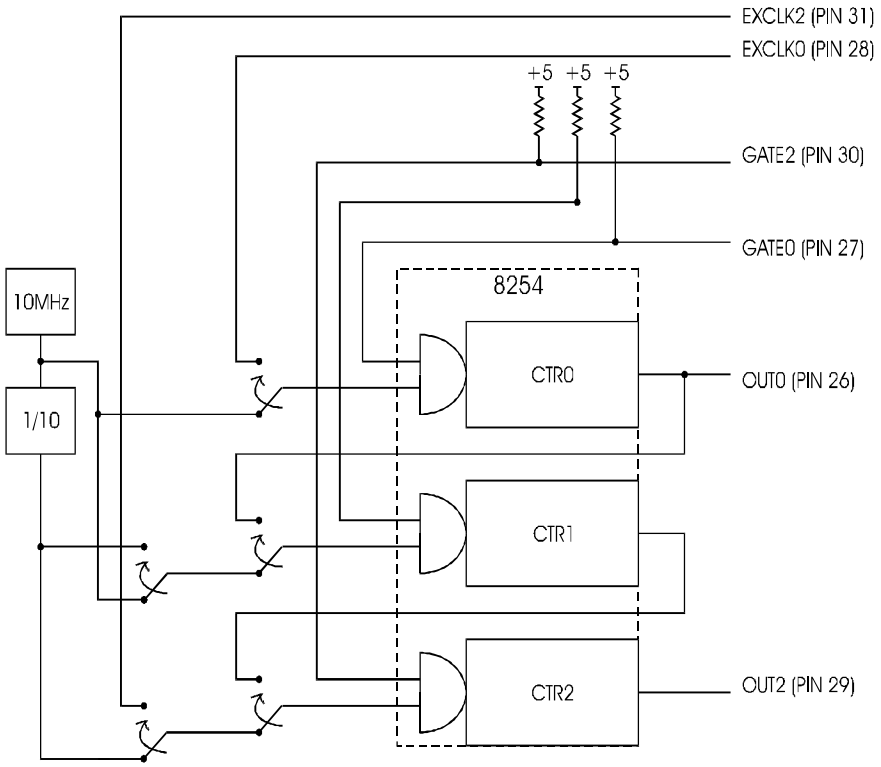
BCD: Set the selected counter to count in Binary Coded Decimal or straight Binary. (BCD=1 selects BCD mode, BCD=0 selects straight binary coding).

Base + 8 Write: Interrupt and Clock Source Control

Because of the limited number of pins on the 32 pin connector, all signals from all three counters are not available externally. This is compensated for by the ability to chain counters and control the count source. Bit 0 and 3 of port C are the outputs of the 82C55 programmed in Mode 1 or Mode 2. These two modes provide an interrupt control of closely coupled parallel interfaces. To program an 82C55 for Mode 1 or 2 you will need to consult the 82C55 data sheet.

Counter Source Control

The counters on the PIDO24 have quite a bit of programmability. Due to the architecture of the board, there is additional counter source and chaining flexibility beyond the standard functions of the 82C54. The diagram shows schematically the programmable options.



The Interrupt and Clock Source Control byte format is as follows:

B7	B6	B5	B4	B3	B2	B1	B0
INT2	INT1	INT0	CKSEL2	CKSEL1	CKSEL0	CLK1	CLK0

INT0-INT2: These bits select the Interrupt Source.

INT2	INT1	INT0	Interrupt Source
0	0	0	No interrupts
0	0	1	External Interrupt
0	1	0	Interrupt from Bit 0 of Port C
0	1	1	Counter 0
1	0	0	Counter 1
1	0	1	Counter 2
1	1	0	Interrupt from Bit 3 of Port C

Clock Source Select

These control register bits select the counter clock source for each counter. Selection of counter cascading other than as three 16-bit counters will override the counter source selection for the linked counters. The clock source bits may be set as follows:

CKSEL0	**Counter 0 Clock Source**
0	10 MHz clock
1	External clock - applied at pin 28

CKSEL1	**Counter 1 Clock Source**
0	10 MHz clock
1	1 MHz clock

CKSEL2	**Counter 2 Clock Source**
0	1 MHz clock
1	External clock - applied at pin 31

Counter Cascading

Counters may be linked into 32 or 48 bit depth. Counter cascading will override any other counter source selection for counters 1 and 2.

CLK1	CLK0	Counter Cascading
0	0	3, 16 bit counters
0	1	1, 16 bit counter (Counter 0) and1, 32 bit counter (Counter 1 cascaded into Counter 2)
1	0	1, 48 bit counter (Counter 0 cascaded into Counter 1 and Counter 2)
1	1	Not Defined

Counter Gates

The counter gates are tied high through a 10K resistor. In this manner the gates are always enabled. Counters 0 and 2 may be disabled by bringing the gates to ground at pin 27 (Counter Gate 0) and pin 30 (Counter Gate 2). Counter Gate 1 has no external access so counter 1 is always enabled.

Reading and Loading the Counters

If you attempt to read an active counter, you will most likely get erroneous data. This is partly caused by carries rippling through the counter during the read operation. Also, the low and high bytes are read sequentially rather than simultaneously and, thus, it is possible that carries will be propagated from the low to the high byte during the read cycle. To circumvent these problems, you should perform a counter-latch operation in advance of the read cycle. To do this, load the RW1 and RW2 Counter Control Bits with zeroes. This instantly latches the count of the selected counter (selected via the SC1 and SC0 bits) in a 16-bit hold register. A subsequent read operation on the selected counter returns the held value. Latching is the best way to read an active counter without disturbing the counting process. You can only rely on directly-read counter data if the counting process is suspended while reading, by bringing the gate low, or by halting the input pulses.

For each counter you must specify in advance the type of read or write operation that you intend to perform. You have a choice of loading/reading (a) the high byte of the count, or (b) the low byte of the count, or (c) the low byte followed by the high byte.

Counter Programming Examples

Using Counter #0 as a Pulse Counter

Note that the counters are "down" counters so, when resetting them, it's better to load them with a full count value of 65,535. In order to read the number of input pulses that have been applied, read back the counter as shown in the following paragraph and subtract that reading from the original loaded value.

```
outportb(BASEADDRESS + 7,0x30);      /* counter 0, mode 0 */
outportb(BASEADDRESS + 4,0xff);      /* counter 0 low load byte */
outportb(BASEADDRESS + 4,0xff);      /* counter 0 high load byte */
```

Reading Counter #0

```
outportb(BASEADDRESS + 7,0x30);/* counter 0, latch command */

/* read in both bytes of the latched value and combine into an integer */

value = inportb(BASEADDRESS + 4) + (inportb(BASEADDRESS + 4) * 256;
```

Generating Square Waves of Programmed Frequency

Frequency of output is a direct function of the frequency of the clock input and of the count loaded into the counter. The minimum count (or divisor) is 2 and the maximum is 65535.

Calculating what divisor to use for a specific output frequency is straightforward. If, for example, you desire a 2000 Hz output and your clock source is 10 MHz, divide it by 2000 and find that the count loaded into counter #0 should be 5000.

Measuring Frequency and Period

The two previous sections show how to count pulses and generate output frequencies. It is also possible to measure frequency by raising the gate input of Counter #0 for a known time interval and counting the number of clock pulses accumulated for that interval. The gating signal can be derived from Counters #1 and #2 operating in a square wave mode.

Counter #0 can also be used to measure pulse width or half period of a periodic signal. The signal should be applied to the gate input of Counter #0 and a known frequency applied to the Counter #0 clock input. During the interval when the gate input is low, Counter #0 is loaded with a full count of 65,535. When the gate input goes high, the counter begins decrementing until the gate input goes back low at the end of the pulse. The counter is then read and the change in counts is a linear function of the duration of the gate input signal. If Counter #0 receives 10 microsecond duration clock pulses (100 KHz), the maximum pulse duration that can be measured is 65,535*10-5 = 655 milliseconds.

Generating Time Delays

There are four methods of using Counter #0 to generate programmable time delays.

Pulse on Terminal Count

After loading, the counter output goes low. Counting is enabled when the gate goes high. The counter output will remain low until the count reaches zero, at which time the counter output goes high. The output will remain high until the counter is reloaded by a programmed command. If the gate goes low during countdown, counting will be disabled as long as the gate input is low.

Programmable One-shot

The counter need only be loaded once. The time delay is initiated when the gate input goes high. At this point the counter output goes low. If the gate input goes low, counting continues but a new cycle will be initiated if the gate input goes high again before the timeout delay has expired; i.e., is re-triggerable. At the end of the timeout, the counter reaches zero and the counter output goes high. That output will remain high until re-triggered by the gate input.

Software Triggered Strobe

This is similar to Pulse-on-Terminal-Count except that, after loading, the output goes high and only goes low for one clock period upon timeout. Thus, a negative strobe pulse is generated a programmed duration after the counter is loaded.

Hardware Triggered Strobe

This is similar to Programmable-One-Shot except that when the counter is triggered by the gate going high, the counter output immediately goes high, then goes low for one clock period at timeout, producing a negative-going strobe pulse. The timeout is re-triggerable; i.e., a new cycle will commence if the gate goes high before a current cycle has timed out.

Customer Comments

If you experience any problems with this manual or just want to give us some feedback, please email us at: ***manuals@accesioproducts.com.***. Please detail any errors you find and include your mailing address so that we can send you any manual updates.

ACCES I/O PRODUCTS, INC.
10623 Roselle Street, San Diego CA 92121
Tel. (619)550-9559 FAX (619)550-7322
www.accesioproducts.com