

## **Model PCL-722**

144 Bit DIO Card

# **PCL-722 144 BIT DIGITAL I/O CARD**

## **USER'S MANUAL**

This documentation is copyrighted 1993 by Advantech Co., Ltd. All rights are reserved. Advantech Co., Ltd. reserves the right to make improvements in the products described in this manual at any time without notice.

No part of this manual may be reproduced, copied, translated or transmitted, in any form or by any means without the prior written permission of Advantech Co., Ltd. Information provided in this manual is intended to be accurate and reliable. However, Advantech Co., Ltd. assumes no responsibility for its use, nor for any infringements of rights of third parties which may result from its use.

PC-LabCard is a trademark of Advantech Co., Ltd. IBM, PC, PC/XT and PC/AT are trademarks of International Business Machines Corporation. OPTO-22 is a trademark of OPTO-22 Corporation.

Part No. 2003722010 Rev.A2  
Printed in Taiwan Aug. 1993

## TABLE OF CONTENTS

<b>CHAPTER 1 GENERAL INFORMATION</b> .....	1
1.1. Introduction .....	1
1.2. Features .....	3
1.3. Applications .....	3
1.4. Electrical Specifications .....	3
1.5. Physical Specifications .....	4
1.6. Power Requirements (PC-XT/AT I/O channel) .....	5
1.7. Transfer Rate .....	5
<b>CHAPTER 2 INSTALLATION</b> .....	7
2.1. Initial Inspection .....	7
2.2. Base Address Setting .....	9
2.3. Address Parameters .....	11
2.4. Interrupt Settings .....	13
<b>CHAPTER 3 OPERATION</b> .....	15
3.1. Configuration .....	15
3.2. Interrupt Handling .....	17
<b>CHAPTER 4 PROGRAMMING</b> .....	21
4.1. Programming Notes .....	21
4.2. Programming Examples .....	22

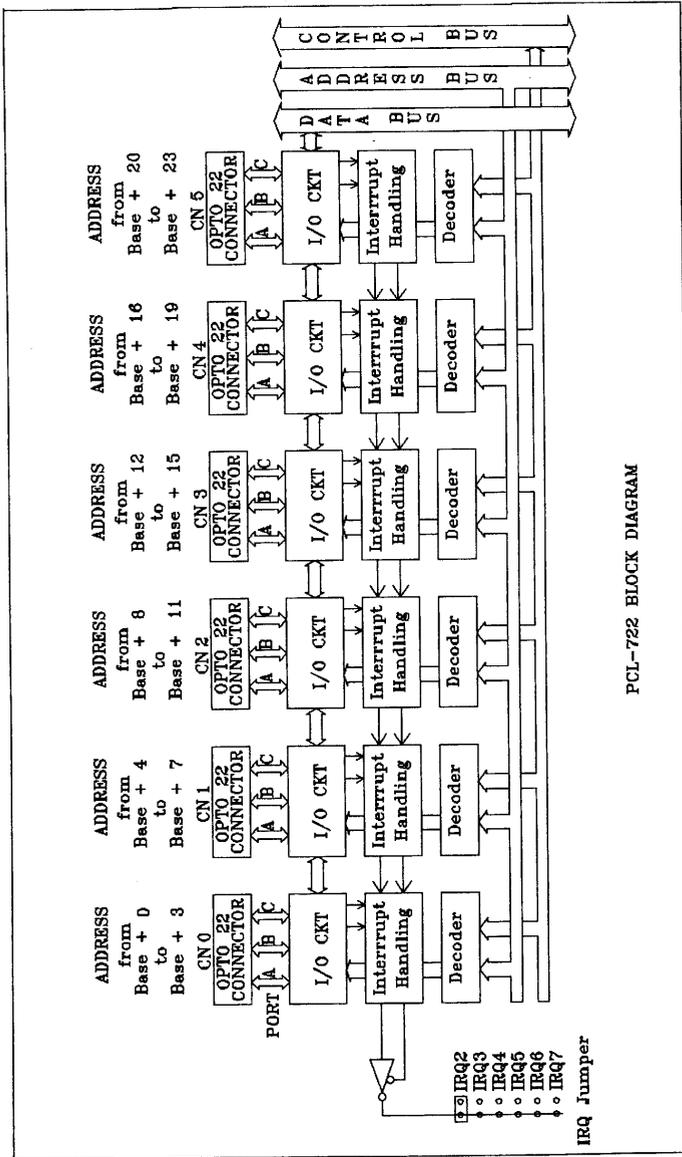
# CHAPTER 1 GENERAL INFORMATION

## 1.1. Introduction

The PCL-722 144 bit DIO card is a generalized 144 line parallel digital I/O board that has six channels. Each channel emulates mode 0 of a 8255 programmable peripheral interface chip, and has 24 lines of digital input and/or output. Every channel has an individual connector with OPTO-22 compatible pin assignment.

Two lines of each connector can be used to generate a hardware interrupt. These inputs are buffered and can be connected to IRQ 2, 3, 4, 5, 6 and 7.

The PCL-722 is a full-size PC add-on card and can fit any full-length expansion slot in the IBM PC XT/AT or compatible computer running at any clock rate.



PCL-722 BLOCK DIAGRAM

## 1.2. Features

- \* 144 TTL digital I/O lines, (6 connectors, each with 24 I/O lines).
- \* The 24 I/O lines of each connector are configured to port PA (8 bit), PB (8 bit), PC Upper (4 bit) and PC Lower (4 bit). Each port can be programmed to digital input or digital output.
- \* Emulates mode O of 8255 PPI
- \* Programmable interrupt handling
- \* OPTO-22 compatible 50-pin connector
- \* Direct interface to wide range of peripherals

## 1.3. Applications

- \* Industrial AC/DC I/O module monitoring and control
- \* Relay and switch monitoring and control
- \* Parallel data transfer
- \* Sensing the signals of TTL, DTL, CMOS logic
- \* Driving indicator LED's

## 1.4. Electrical Specifications

### a. Input signal specifications

Input logic high voltage : 2.0 V -- 5.25 V  
Input logic low voltage : 0.0 V -- 0.80 V  
High level input current : 20.0 uA  
Low level input current : - 0.2 mA

b. Output signal specifications

Port A and Port B

output logic high voltage : 2,4 V (min)  
output logic low voltage : 0.5 V (max)  
High level output current : -2.6 mA  
Low level output current : 24.0 mA

Port C

output logic high voltage : 2.4 V (min)  
output logic low voltage : 0.4 V (max)  
High level output current : -15.0 mA  
Low level output current : 24.0 mA

c. Operating environment

Temperature : 0 °C to 70 °C  
Relative Humidity : 5% to 90% noncondensing

d. Storage environment

Temperature : -55 °C to 150 °C  
Relative Humidity : 5% to 90% noncondensing

**1.5. Physical Specifications**

1. Dimensions : 333.5 mm by 99.06 mm
2. I/O connector : 50 pin male ribbon cable connector

## **1.6. Power Requirements (PC-XT/AT I/O channel)**

Typical power : 1.3 A at 5V DC(+ 5%)

Maximum power : 1.8 A at 5V DC(+ 5%)

## **1.7 Transfer Rate**

Typical : 300 K bytes/sec

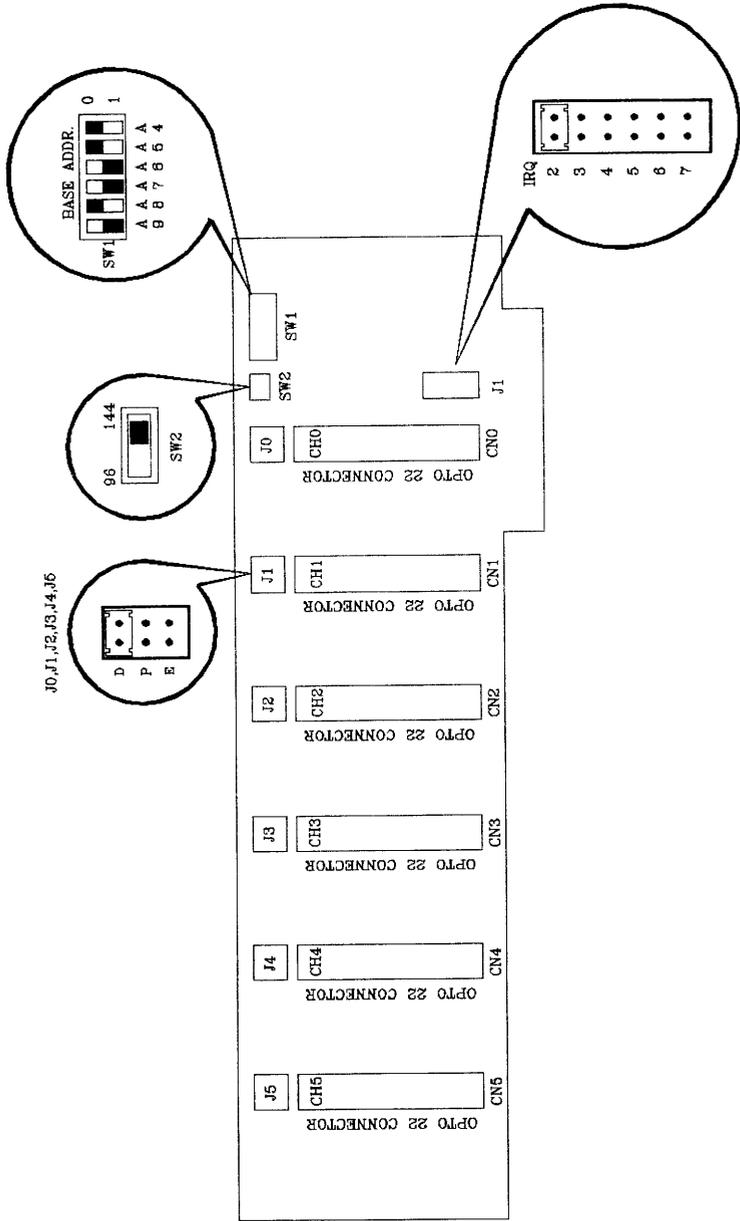
Maximum : 500 K bytes/sec

## **CHAPPER 2 INSTALLATION**

### **2.1. Initial Inspection**

#### **CAUTION !**

Do not install or remove the PCL-722 I/O board while the power is ON, as this may damage the plug-in board, computer, or both. Always ensure that the power has been turned OFF before proceeding with installation or removal.



## CONNECTOR PIN ASSIGNMENT

PC 7	1	2	GND
PC 6	3	4	GND
PC 5	5	6	GND
PC 4	7	8	GND
PC 3	9	10	GND
PC 2	11	12	GND
PC 1	13	14	GND
PC 0	15	16	GND
PB 7	17	18	GND
PB 6	19	20	GND
PB 5	21	22	GND
PB 4	23	24	GND
PB 3	25	26	GND
PB 2	27	28	GND
PB 1	29	30	GND
PB 0	31	32	GND
PA 7	33	34	GND
PA 6	35	36	GND
PA 5	37	38	GND
PA 4	39	40	GND
PA 3	41	42	GND
PA 2	43	44	GND
PA 1	45	46	GND
PA 0	47	48	GND
+5V	49	50	GND

### 2.2. Base Address Setting

The PCL-722 requires 32 consecutive address locations within the I/O space. Some I/O address locations will be occupied by internal I/O and your peripherals. In order to avoid conflict with these devices, the PCL-722 address can be set by a 6 position DIP switch and placed anywhere in the IBM PC decoded external UO space.

The 200-3FF hex (512-1023 decimal) address range provides extensive unused areas of I/O space, though you should check with the IBM Standard Technical Reference Manual for possible conflicts with commonly installed peripherals.

144 LINES OF I/O

BASE ADDRESS SWITCH SETTING EXAMPLES (In hex)

<p>SW1 2C0 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>	<p>SW1 200 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>	<p>SW1 3A0 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>
<p>SW1 2C0 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>	<p>SW1 200 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>	<p>SW1 3A0 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>

Note: While selecting 144 lines, A4 is not in functional status.

96 LINES OF I/O

BASE ADDRESS SWITCH SETTING EXAMPLES (In hex)

<p>SW1 2C0 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>	<p>SW1 290 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>	<p>SW1 2F0 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>
<p>SW1 310 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>	<p>SW1 380 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>	<p>SW1 3E0 ON OFF 0 1 1 2 3 4 5 6 A A A A A A 9 8 7 6 5 4</p>

I/O ADDRESS	DEVICE	I/O ADDRESS	DEVICE
000-0FF	System I/O	380-38F	SDLC 2
200-207	Game I/O	3B0-3BF	Monochrome card
278-27F	LPT 2	3C0-3CF	EGA card
2F8-2FF	COM 2	3D0-3DF	Color graphics
300-31F	Prototype card	3F0-3F7	Floppy disk
378-37F	LPT 1	3F8-3FF	COM 1

Fig. 2.1 IBM PC-XT/AT I/O map (in hex)

### 2.3. Address Parameters

The PCL-722 can limit the 32 required addresses to 16, therefore, the digital I/O is limited to 96 bits. SW2 (see Fig. 2.2 below) sets the number of useable I/O. The PCL-722 card should be left in the 144 bit mode unless this address conflicts with the addresses of other peripherals already installed or those to be installed in neighboring slots.

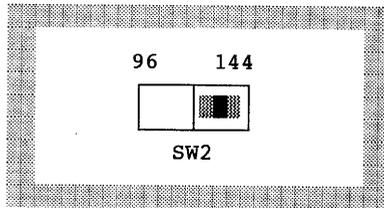


Fig. 2.2 Select 144 or 96 I/O

By selecting 96 bit mode, two or more boards can be mapped consecutively on adjacent addresses. By selecting 144 bit mode, a minimum gap of 8 addresses will exist between each board. The following address list provides examples of multiple board addressing with consecutive addresses. Note that these are not the only addresses available because other combinations can be used.

Example :

Board 1	Board 2
-----	-----
200-20F	210-21F
230-23F	240-24F
250-25F	260-26F

Note : In 144 bit mode, base address line A4 is not in functional status, therefore, A4 of SW1 does not need to be set.

The following base addresses are available:

200,220,240,260,280,2A0,2C0,2E0  
 300,320,340,360,380,3A0,3C0,3E0

The following base addresses are illegal:

210,230,250,270,290,2B0,2D0,2F0  
 310,330,350,370,390,3B0,3D0,3F0

You can set the address according to the information given in the following table.

I/O ADDRESS RANGE	SWITCH POSITION (SW1)					
(HEXADECIMAL)	1	2	3	4	5	6
	A9	A8	A7	A6	A5	A4
200 - 20F	1	0	0	0	0	0
220 - 22F	1	0	0	0	1	0
...						
300 - 30F	1	1	0	0	0	0
...						
3E0 - 3EF	1	1	1	1	1	0

NOTE: 0 = ON, 1 = OFF

A4 through A9 correspond to your PC's address lines.

### 2.4. Interrupt Settings

Pins PC-0 and PC-3 of each channel of the PCL-722 generate hardware interrupts and can be set to any of the interrupt request lines IRQ2, 3, 4, 5, 6 and 7.

The IRQ signal is TRI-STATED allowing multiple PCL-722 boards to share the same interrupt line. For example, two PCL-722 boards in a system can have IRQ2 selected.

The following diagram shows the locations of interrupt enable selection jumpers .

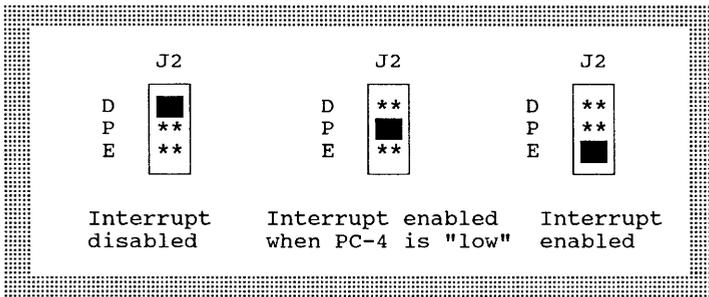


Fig. 2.3 Interrupt Enable selection

## CHAPTER 3 OPERATION

The PCL-722 card emulates MODE 0 of 8255 PPI, and is pin compatible with popular industrial solid state I/O racks and modules such as those manufactured by OPTO-22, Potter Brumfield, Gordos, and others.

The PCL-722 has six 50-pin male IDC connectors that interface with position OPTO-22 racks 8, 16 and 24.

Two of the I/O lines (PC-0, PC-3) of each channel can be used to generate a hardware interrupt.

Note : The PCL-722 emulates MODE 0 of 8255 PPI, therefore, it is in effect the same as a 8255 controller card and all packages or programs that support the 8255 in MODE 0 can be used on PCL-722 without reconfiguring software.

### 3.1. Configuration

Mode 0 of 8255 provides simple input and output for each of the three ports and no handshaking is required as data is written directly to or read from a specified port (see 8255 MODE 0 Function Definitions).

Read-back is another useful function of the PCL-722 and this allows the user to monitor each output port of each channel.

#### 8255 MODE 0 FUNCTION DEFINITIONS

- Two 8-bit ports (Port A, Port B)
- Two 4-bit ports (Port C upper, Port C lower)
- Any port can be used for input or output
- Outputs are latched whereas inputs are not latched
- 16 different input/output configurations are available in this mode.

Register	Address	Function
Port A	BASE ADDRESS + 0	Read/Write
Port B	BASE ADDRESS + 1	Read/Write
Port C	BASE ADDRESS + 2	Read/Write
CFG REG.	BASE ADDRESS + 3	Write Only

Fig. 3.1 8255 address map

EXAMPLE: Base address set at 300

	CH0	CH1	CH2	CH3	CH4	CH5
Port A	300	304	308	30C	310	314
Port B	301	305	309	30D	311	315
Port C	302	306	30A	30E	312	316
CFG Port	303	307	30B	30F	313	317

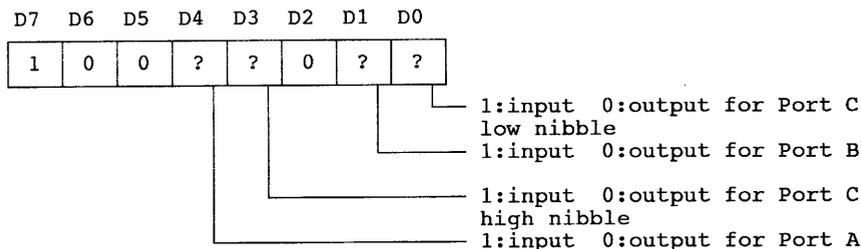


Fig. 3.2 CFG register format (mode 0)

The following table lists all possible configured "commands" to configure 8255 in mode 0.

8255 Mode 0 Configuration Table

	D4	D3	D1	D0
Config.	PA0 - PA7	PC4 - PC7	PB0 - PB7	PC0 - PC3
80H	output	output	output	output
81H	output	output	output	input
82H	output	output	input	output
83H	output	output	input	input
88H	output	input	output	output
89H	output	input	output	input
8AH	output	input	input	output
8BH	output	input	input	input
90H	input	output	output	output
91H	input	output	output	input
92H	input	output	input	output
93H	input	output	input	input
98H	input	input	output	output
99H	input	input	output	input
9AH	input	input	input	output
9BH	input	input	input	input

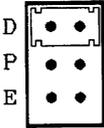
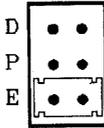
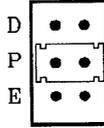
### 3.2. Interrupt Handling

The PCL-722 has two lines (PC-0, PC-3) for each channel to generate hardware interrupt.

The interrupts are edge-triggered. PC-0 traps a rising edge signal, while PC-3 traps a falling edge signal. To use the interrupts, simply insert the appropriate IRQ jumper to select the desired interrupt level (IRQ 2-7).

To determine which input generated the interrupt, read Port C lower, bits 0 and 3, to locate the current state of inputs. Note that the inputs are not latched and no "first event" trapping is provided to determine which input was active first.

The user can select which channel is to be interrupt enable by setting interrupt enable jumpers J0, J1, J2, J3, J4 and J5. Refer to the table below :

<div style="border: 1px solid black; padding: 10px; width: fit-content; margin: auto;">NO INTERRUPT REQUEST</div>	 <p style="margin-top: 5px;">Interrupt disabled</p>																	
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="2">BUS STATUS</th> <th rowspan="2">RESPONSE</th> </tr> <tr> <th>PC-0</th> <th>PC-3</th> </tr> </thead> <tbody> <tr> <td>H</td> <td>X</td> <td>NO INTERRUPT REQUEST</td> </tr> <tr> <td>L</td> <td>H-&gt;L</td> <td>INTERRUPT REQUEST</td> </tr> <tr> <td>X</td> <td>L</td> <td>NO INTERRUPT REQUEST</td> </tr> <tr> <td>L-&gt;H</td> <td>H</td> <td>INTERRUPT REQUEST</td> </tr> </tbody> </table>	BUS STATUS		RESPONSE	PC-0	PC-3	H	X	NO INTERRUPT REQUEST	L	H->L	INTERRUPT REQUEST	X	L	NO INTERRUPT REQUEST	L->H	H	INTERRUPT REQUEST	 <p style="margin-top: 5px;">Interrupt enabled</p>
BUS STATUS		RESPONSE																
PC-0	PC-3																	
H	X	NO INTERRUPT REQUEST																
L	H->L	INTERRUPT REQUEST																
X	L	NO INTERRUPT REQUEST																
L->H	H	INTERRUPT REQUEST																
	 <p style="margin-top: 5px;">* if PC-4 'Low', Interrupt enabled</p> <p style="margin-top: 5px;">* if PC-4 'High', Interrupt disabled</p>																	

Note : In the event that an interrupt signal is generated from a rising-edge or a falling-edge change, and if additional interrupt signals are still to be inputted, the bit-status must be reset to its original station, i.e. PC-0 should be set at 0 and PC-3 should be set at 1 before such signals can be inputted. For example, lxx0 (1000, 1010, 1100, 1110)

Although interrupt signals are normally received from external peripherals, the PCL-722 can also execute a (test) output signal to simulate an interrupt being inputted from an external device. See the example in next chapter.

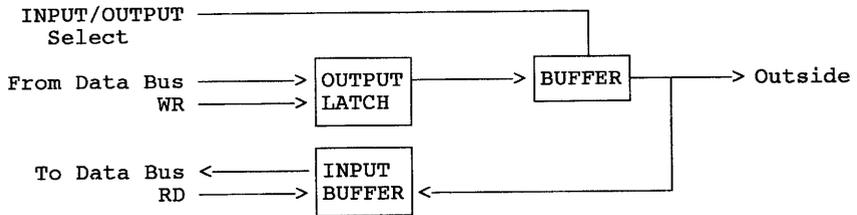
# CHAPTER 4 PROGRAMMING

## 4.1. Programming Notes

The signal direction of PCL-722 can be INPUT or OUTPUT by software programming. However, after power-on or hardware (front panel RESET button of the PC is pressed) reset, all the ports of the PCL-722 are automatically configured as INPUT direction.

The user can OUTPUT data to the port which is configured as INPUT without affecting the input data. The output data is latched internally and not transmitted to the part until the port is configured as OUTPUT.

It is recommended that before initiating the direction configuration of the PCL-722, be sure to set the wanted initial output status first. This means that if you want the initial output data of port N is 'AD'(in hex,), FIRST output port N the data ('AD' here), then configure port N as output function. (Refer to the following examples for details.)



1. The output driver is disabled (high impedance) when the port is set to INPUT or after RESET.
2. The output data written to the port is latched in the OUTPUT DATA LATCH no matter the port is set as INPUT or OUTPUT.
3. When the port is set as OUTPUT, the input circuit is used as output status read-back.

4. The data in the OUTPUT DATA LATCH is in a random status after power on. It must be initialized before setting the port to output direction.

## 4.2. Programming Examples

The PCL-722 is easily programmed and the following section lists some program examples. All the program examples below have been fully tested on PC-XT/AT or compatible computers.

Note : The following programs should be run under Turbo C version 1.5 or 2.0.

### DEMO1.C :

```

/*****\
 * This demo program demonstrates how to use the read back *
 * function of the PCL-722 to monitor the output status. *
 * HARDWARE SETTING : *
 * 1. Base address set at 0x2C0 *
 * 2. select 144 lines I/O (by adjusting SW2) *
 \*****/

#include <stdio.h>
#include <conio.h>
#include <process.h>
#include <dos.h>

struct channel /* define channel structure */
{
    int portA;
    int portB;
    int portC;
    int config;
}

main()
struct channel CN[6]; /* there are six connectors */
int base = 0x2c0; /* set base address at 2C0 (in hex) */
int portA; /* save read back value of port A */
int portB; /* save read back value of port B */
int portC; /* save read back value of port C */
int i, j;

/* screen handle x/

```

```

clrscr();
gotoxy(25,1);
textattr(0x70);
cputs("PCL-722 DEMO PROGRAM");
gotoxy(1,2);
printf("Testing CN : ");
for (i=0;i<3;i++)
  { gotoxy(1,4+i*2);
    printf("Port%c output value --> ", 'A'+i);
    gotoxy(40,4+i*2);
    printf("ReadBack -->");
  }

/* calculate port addresses of each CN */

for (i=0;i<6;i++)
  { CN[i].portA = base + i*4 + 0;
    CN[i].portB = base + i*4 + 1;
    CN[i].portC = base + i*4 + 2;
    CN[i].config= base + i*4 + 3;

/* Initialization */

    outportb (CN[i].portA , 0x00);
    outportb (CN[i].portB , 0x00);
    outportb (CN[i].portC , 0x00);
    outportb (CN[i].config, 0x80); /* all as OUTPUT */
  }

/***** main program *****/

for(i=0;i<6;i++)
  { gotoxy(13,2); /* */
    textattr(0x09); /* print CN number */
    cprintf("%d",i); /* */

    for(j=0;j<0x100;j++)
      { outportb ( CN[i].portA, j); /* out j to port A */
        gotoxy(25,4);
        printf("%2x",j);
        portA = inportb (CN[i].portA); /* read back */
        gotoxy(53,4);
        printf("%2x",portA);
        if ( portA != j)
          { printf("\7"); /* beep */
            gotoxy(1,20);
            textattr(0x09);
            cprintf("CN %d port A readback error!",i);
            getch();
            exit(1); /* quit to dos */
          }
        outportb ( CN[i].portB, j); /* out j to port B */
        gotoxy(25,6);
      }
  }

```

```

printf("%2x",j);
portB = inportb (CN[i].portB); /* read back */
gotoxy(53,6);
printf("%2x",portB);
    if ( portB I= j)
        { printf("\7"); /* beep */
          textattr(0x87);
          gotoxy(1,20);
          cprintf("CN %d port B readback error!",i);
          getch();
          exit(1); /* quit to dos */
        }
    outportb ( CN[i].portC, j); /* out j to port C */
    gotoxy(25,8);
    printf("&2x",j);
    portC = inportb (CN[i].portC); /* read back x/
    gotoxy(53,8);
    printf("%2x",portC);
        if ( portC != j)
            { printf("\7"); /* beep */
              gotoxy(1,20);
              textattr(0x87);
              cprintf("CN %d port C readback error!",i);
              getch();
              exit(1); /* quit to dos */
            }
    }
}
printf("\n\nPass test ! Press any key to quit...\n");
getch();
}

```

## DEMO2.C

```

/*****\
* This demo program demonstrates how to use the capability *
* of interrupt handling of the PCL-722 card. *
* *
* HARDWARE SETTING : *
* In this program, the BASE ADDRESS is set at '0x2CO', *
* and the interrupt level is set at IRQ2 ( associated to *
* INT OAH ISR ), and all the interrupt setting jumpers *
* of each channel are set at the 'P' position, i.e. the *
* interrupt request of each channel can be programmed to *
* be enabled or disabled. *
* PROGRAM DESCRIPTION : *
* In this program, the original INT OAH ISR is replaced *
* by a new ISR named 'alarm'. If a hardware interrupt *
* signal is generated, the PC will beep.(alarm) *
/*****\

```

```

#include <dos.h>
#include <stdio.h>
#include <conio.h>

/* new INT OAH ISR */

void interrupt alarm (void)
{
    char Port61_Old_Status;
    char Port61_New_Status;
    int DelayTime = 0x300;
    int Count = 0;
    int i;

    /** get original port 61H status, and save it **/

    Port61_New_Status = Port61_Old_Status = inportb(0x61);

    /** set port 61h bit1 = '0' **/

    Port61_New_Status &= 0xfd;    /* 1111 1101 in binary */

    for ( Count=0 ; Count<0x300 ; Count++ , DelayTime-- )
        {
            Port61_New_Status ^= 0x02;    /* On/Off bit1 */
            outportb (0x61 , Port61_New_Status);
            for (i=0 ; i<DelayTime ; i++);
        }
    outportb (0x61 , Port61_Old_Status);
    outportb (0x20, 0x20);    /* send EOI to 8259 */
}

void main()
{
    int IMR;
    void interrupt (*Int_A_Old_Vector)();
    Int_A_Old_Vector = getvect(0x0a);    /* get old int Oah ISR */
    setvect (0x0a , alarm);    /* set new int Oah ISR */
    IMR = inportb(0x21);    /* get 8259 interrupt mask register */
    /* initialization */

    outportb (0x2c0, 0x0);    /* CNO, port A */
    outportb (0x2c1, 0x0);    /* CNO, port B */
    outportb (0x2c2 , 0x08);    /* CNO,PC4=0-->interrupt enable */
                                /* PC3=1 */
                                /* PC0=0 */
    outportb (0x2c3 , 0x80);    /* set CNO as output function */

    clrscr();
    printf("Press any key to generate a rising edge interrupt");
}

```

```
getch();
outportb(0x21,0xfb & IMR);          /* set IRQ2 nonmasked */

printf("\n\nCN0 -- Port C = %x",inportb(0x2c2));
/* generate a rising edge signal at CNO, PCO */
outportb (0x2c2 , 0x9);

printf("\n\nCN0 -- Port C = %x",inportb(0x2c2));
printf("\n\npress any key to continue...");
getch();
clrscr();
printf("Press any key to generate a falling edge interrupt");
getch();
outportb(0x21,0xfb & IMR);          /* set IRQ2 nonmasked */

outportb (0x2c2 , 0x08); /* CNO,PC4=0-->interrupt enable */
                        /* PC3 = 1 */
                        /* PCO = 0 */
printf("\n\nCN0 -- Port C = %x",inportb(0x2c2));
/* generate a falling edge signal at CN)O, PC3 */

outportb (0x2c2 , 0x0);

printf("\n\nCN0 -- Port C = %x",inportb(0x2c2));
printf("\n\npress any key to quit...");
getch();

/* set CNO as INPUT to release IRQ2 line */

outportb(0x2c3,0x9b);

/* restore old INT OAH ISR */

setvect (0x0a , Int_A_Old_Vector);

outportb(0x21,IMR); /* restore 8259 interrupt mask */
}
```

## PCLD-7224Q USER'S NOTE

To use the PCL-722 144 bit DIO card with the PCLD-7224Q, you have to apply a 3 - 8 volts of external DC power into the " + " and "-" terminal of TBP. The DC power would supply to the logic side of the SSR module.

PCLD-7224Q also allows you to use the +5V DC power of PC. In this situation, you have to short the FVCC besides the TBP by a conductor or fuse like the F1.

