PC-62B and PC-65

User's Manual

October 1991 Edition

First edition.

October 1991

October 1991 Printing

1

Introduction

Features

- Sixteen channel opto-isolated inputs or outputs.
- Wide input/output voltage range.
- Single ended or differential inputs/outputs.
- Jumper selectable wait state generator allows trouble free operation in all high speed systems.
- Driver software supplied, complete with C and BASIC source code.
- Includes Lab Windows drivers.

Overview

The PC-62B and PC-65 optically isolated input and output boards are designed to counter problems in interfacing and control with electrical interference and signals referenced to dissimilar grounds. There are sixteen input (PC-62B) or output (PC-65) digital lines, each isolated with a 4N25 opto-isolator. The isolation voltage of the opto-isolators is 2.5kV for up to one minute. The individual lines can be kept completely separate from one another, or they may use a common return line. This makes the opto-isolator boards very flexible for use in environments (especially industrial) where signals may be floating or referenced to a voltage other than the system ground.
The board plugs into any fully bussed slot of an IBM PC/XT/AT or PS/2 model 25 or 30 or any compatible ISA or EISA machine, including 8088, 286, 386 or 486 based systems. It occupies

two consecutive I/O addresses. The base I/O address can be set in the range 0h to 7F8h with a DIP switch on the board.

The PC-62B and PC-65 feature wait state generation circuitry. This ensures that the host computer's I/O bus cycle is long enough for the devices on the opto-isolator board. Zero, 1, 2, 4 or 8 wait states are selectable with a jumper block on the board.

The power supplies +12V, +5V and ground are available on the external user connector.

2

Installation

Configuration

There are three aspects of the PC-62B and PC-65 that can be configured. These are:

- The base address. This is the address where the computer will find the board. This is set to an address between 0h and 7F8h with the eight switches on the DIP switch block. Note that this address is in the computer's I/O address space.

- Wait state generation. If the host computer generates abnormally fast I/O bus cycles then it may be necessary to slow these down when the computer accesses the PC-62B or PC-65. The boards can do this without affecting any other I/O bus cycles. The number of additional wait states inserted in a PC-62B or PC-65 bus cycle is controlled by jumper on the board.

- Single ended or differential input and output. The inputs and outputs of each port may be used totally independently of one another or one side of each line may be connected to a common return line. A common line simplifies connections to the board. The input/output mode for each line is controlled independently, with two blocks of mini-jumps on the board.

Setting the base address

The opto-isolator board uses the first two addresses in a block of eight I/O addresses, one for each eight bit port. The base address setting controls the address where this block begins. This must be set so that the PC-62B or PC-65 does not use any addresses that are used by another device or card. If more than one opto-isolator card is to be installed in the computer then each card must have a different base address setting.
The base address may be assigned to any location from 0h to 7F8h, on eight byte boundaries. Refer to table 1 below for a guide on addresses used by standard I/O devices and table 2 in the appendices for a list of the base address switch settings.
The base address is set by adjusting the eight switches in the DIP switch block on the board. Switch number 1 is used to compare address line A10, number 2 for A9 and so on up to switch 8 which is used to compare address line A3. In general, if a switch is set to the off position, then its corresponding address weighting contributes to the base address. See figure 1 for the DIP switch weightings.

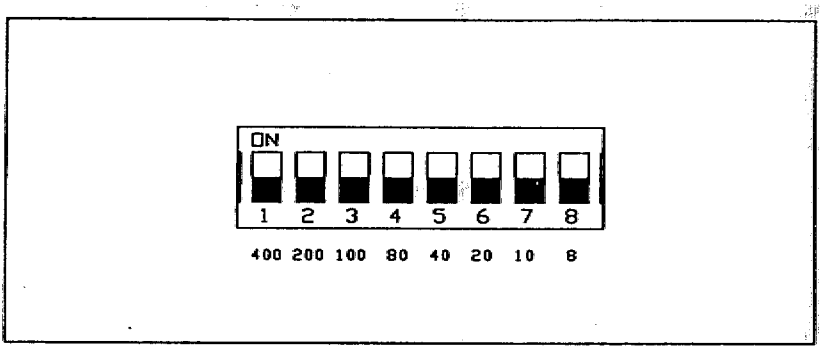

Figure 1: DIP Switch Address Weights (in hex).

For example switches 2 (corresponding to A9) and 4 (corresponding to A7) off yield a base address of 200h + 80h = 280h.

I/O Addresses from 400h to 7FFh

Note that addresses from 400h to 7FFh cannot normally be used because these addresses are not normally decoded by other I/O devices and cards in the 0h to 3FFh range.

Address (hex)	Standard Device
000-1FF	Internal system board
200-20F	Games port
210-217	Expansion unit
220-24F	Reserved
250-257	Not assigned
258-25F	Intel "Above Board"
260-277	Not assigned
278-26F	Reserved
280-2EF	Not assigned
2F0-2F7	LPT2
2F8-2FF	COM2
300-31F	Prototype board
320-32F	Hard disk
330-377	Not assigned
378-37F	LPT1
380-38F	SDLC communications
390-39F	Not assigned
3A0-3AF	Binary communications
3B0-3BF	Monochrome display adapter
3C0-3CF	Reserved
3D0-3DF	Colour graphics adapter
3E0-3E7	Reserved
3E3-3EF	Not assigned
3F0-3F7	Floppy disk
3F8-3FF	COM1
400-7FF	Not used; refer to text

Table 1: Standard I/O Addresses

The PC-62B and PC-65 however (and most other members of the PC-XX family) can use these additional addresses if (and only if) there is no board or device at address 400h less than the address of the PC-62B or PC-65 or the board at the address 400h less also decodes the extra addresses. For example a PC-62B may be installed at address 300h and another PC-62B or a PC-65 at address 700h. However it would not be advisable to install a board at address 778h because the printer port LPT1 uses a base address of 378h and does not normally decode the extra addresses. Most other members of the PC-XX family of boards decode the extra addresses and may safely be installed 400h locations apart.
If your computer has boards not listed in table 1 installed (such as LAN adapters, back-up boards or other engineering boards) then consult the manuals for these boards for information on the address ranges used. In most cases a base address of either 280h or 300h is a good choice. Address 280h is also the factory default address.

Generating additional wait states

The number of additional wait states inserted in I/O bus cycles addressing a PC-62B or PC65 is controlled by the position of the wait state jumper JP1. This is marked 'Wait State Selection' on the board. Refer to figure 2 for the wait state jumper positions. The factory default setting is for zero wait states.

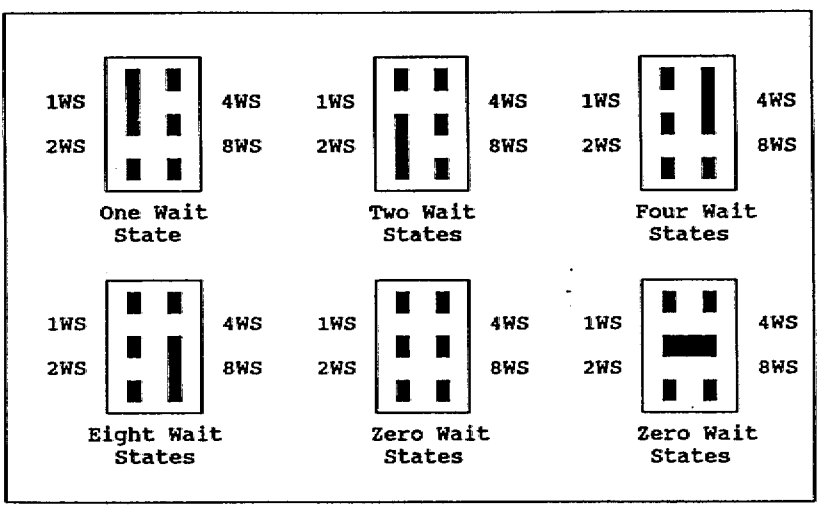

Figure 2: Wait State Jumper Settings

Only a very small number of computers will require additional wait states. If the opto-isolator card in your computer seems to be returning incorrect readings then increase the number of wait states until correct results are obtained. If the card does not produce correct results with the

maximum number of additional wait states inserted then either the host computer or the card is faulty and should be serviced.

Input/output mode

PC-62B, opto-isolated inputs

Each of the input lines of each port may be configured for either single ended or differential input. In differential mode each bit of the two input ports has an input line with a corresponding return line. In single ended mode the return lines are connected to a single common return line. Differential or independent mode is useful where inputs of opposite polarity are to be sampled, the inputs are floating or not referenced to a common ground, or must be isolated from one another.

Single ended mode is useful for signals all referenced to the same voltage, such as a common plant ground. It reduces the number of connections that have to be made to the PC-62B. With all lines in this mode, the PC-62B is hardware compatible with the older PC-62.

There is one common return line per input port. For each bit that is intended to be in single ended mode, a jumper cap is placed over the corresponding pins on the input mode jumper blocks. They are JP3 for port A and JP2 for port B. See figure 3 for a schematic of an input port.

The factory default setting is all jumpers installed for single ended mode.

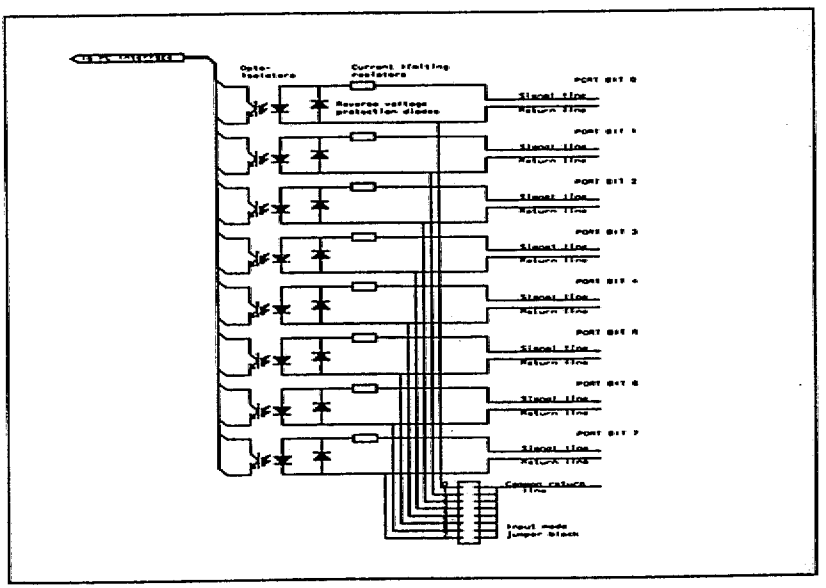

Figure 3: Schematic of an input port

PC-65, opto-isolated outputs

Each of the output bits of each port may be configured for either single line or dual line output. In dual line mode each bit of the two input ports has a signal line with a corresponding return line. In single ended mode the return lines are connected to a single common return line for each port.

Dual line mode is useful where the devices that the PC-65 will control are not referenced to the same ground, are floating or must be isolated from one another.

Single line mode is useful for controlling devices returned to the same ground. It reduces the number of connections that have to be made to the PC-65

There is one common return line per input port. For each bit that is intended to be in single line mode, a jumper cap is placed over the corresponding pins on the output mode jumper blocks. They are JP3 for port A and JP2 for port B. See figure 4 for a schematic of an output port.

The factory default setting is all jumpers installed for single ended mode.

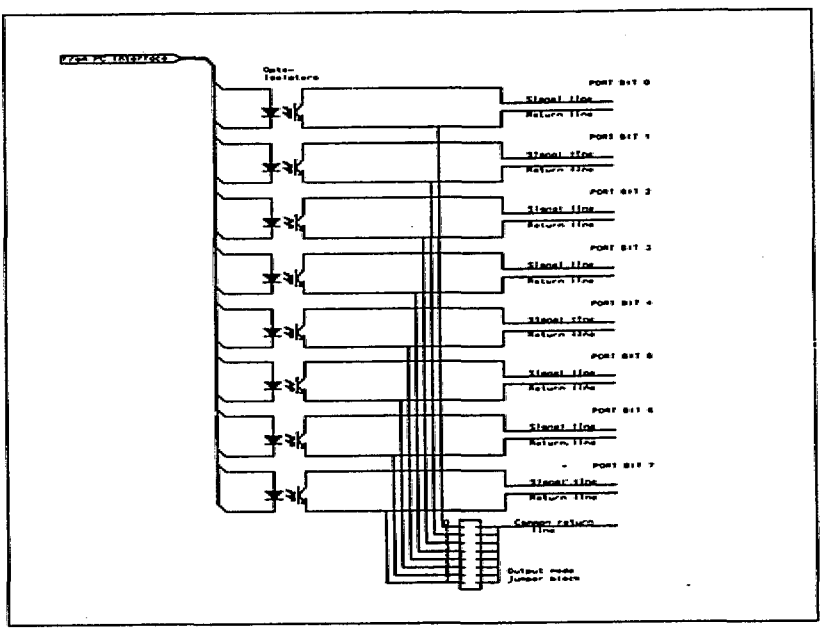

Figure 4: Schematic of an output port

Installation

Installing the opto-isolator card is straightforward. You will need a screwdriver to match the screws on the computer's cover and expansion slot bracket.

a) Switch off the computer and all attached devices.

b) Unplug the power cord from the computer and all attached devices. Failure to do this may result in hazardous conditions, as there may be dangerous voltage levels present on externally connected cables.

c) Remove the cover from the PC. If you are not sure how to do this, consult the manual supplied with the system unit.

d) Choose any unused expansion adapter slot and remove the screw from the top of the blank bracket corresponding to the chosen slot. Remove the bracket.

e) Align the gold plated edge connector of the opto-isolator card with the edge socket on the computer system board and align the board bracket with the rear adapter slot on the PC case. Firmly press the board down into the edge socket. Ensure that the board's edge connector is seated in the socket and has not slipped sideways past the socket.

f) Refit the bracket's screw and replace the computer's cover.

g) Plug in all cords and cables. Switch on the power. The opto-isolator board is now installed.

3

Interconnections

The opto-isolator board plugs into any of the computer's expansion slots, along the board's gold plated edge connector. The board communicates to the external world via a connector mounted in its bracket. This chapter describes these two connectors.

Connections to the computer backplane

The opto-isolator board may be plugged into any slot of the computer backplane (with the exception of the J8 slot of IBM XT's). All communication to and from the host computer is via this connector. The board forms part of the processor's I/O space.
If the opto-isolator board sometimes behaves erratically then there is the possibility that the gold plated contacts on the edge connector may have become dirty or abraded, especially if the card is installed and removed many times from various different computers. This condition can be corrected simply by cleaning the contacts with an ordinary pencil eraser.

User connector

The opto-isolator card's interface to the external world via a 37 way D-type female connector mounted in the board's bracket. It carries the following signals:

- Port A and Port B input lines (PC-62B) or output lines (PC-65).
- Port A and Port B's independent return lines.
- Port A and Port B's common return line.
- Power supplies of +5V, +12V and digital ground.

Figure 5 shows these connections, together with their pin assignments. Note that the pin connections refer to the D-type pin numbers. These are embossed onto the connector itself, and are visible with the board installed the computer.
Input and output connection schemes are discussed separately for the PC-62B and PC-65 below.

PC-62B input connection schemes

Looking into a single bit of an input port, the external signal sees a diode. If the forward voltage applied to the diode is greater than about 3 volts, the diode will conduct and the corresponding bit in the port will reflect a 1 to the host computer. If the voltage applied between the bit's input lines is less than about 3 volts then the computer will interpret the input as a 0. See figure 3 for a diagram of an input port. The minimum operating current for the input diode is 6mA. The inputs are protected against reverse voltages (up to 400V) and are current limited. If a reverse voltage is applied, the protection diode will clamp the reverse voltage to 0.7V. If the input mode jumper is installed for the bit, then the input signal may be applied between the bit's input line and either its return line or the port's common return line. If this jumper is not installed then the signal must be applied between the bit's input line and corresponding return line.
Note that it is the voltage drop across the diode which determines whether the bit is high or low and not the absolute voltage present on the signal line.

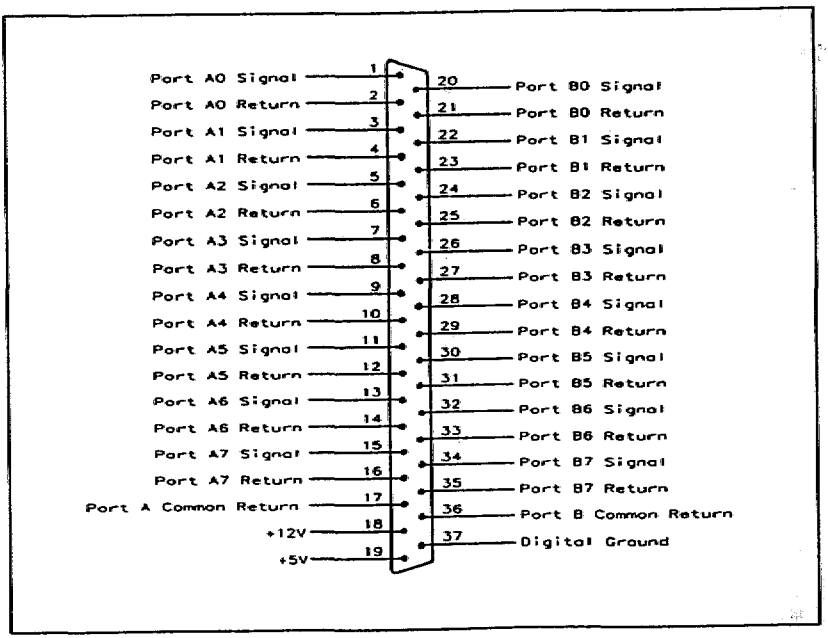

Figure 5: Opto-isolator board connector, as seen from the rear of the PC.

PC-65 output connection schemes

The output bits of each port of the PC-65 are simply opto-isolated NPN transistor switches. When the host computer writes a 1 to a bit in a port its transistor switches on, and a 0 switches the transistor off. The collector of the transistor is connected directly to the corresponding bits output signal line and the emitter directly to the bit's return line. If the corresponding bit has its output mode jumper installed then its emitter is also connected to the port's common return line. See figure 4 for a diagram of an output port. Because of the many different ways of using the output ports for interfacing, current limiting and reverse voltage protection are not provided on the port's outputs. This allows greatest flexibility for interfacing.

Power supplies

The +12V and +5V power supplies are available, together with digital ground, from the connector. These can be used to power external circuitry. Note that no more than about 200mA should be drawn from these power supplies or the opto-isolator board may be damaged permanently.

It is not recommended that the power supplies be used to reference any of the devices that must be isolated from the computer as the power supplies themselves are not isolated. Doing this would defeat the purpose of the opto-isolation. A similar consideration is valid for the ground line.

4

Programming Guide

This chapter describes programming the opto-isolator boards at the lowest level. While this is very straightforward, it requires some knowledge of the system hardware. As an alternative, the driver software supplied with the board can be used. Then the user need not read this section and can proceed directly to Chapter 5: 'Software'. Advantages of using the supplied drivers are:

- Detailed knowledge is not required of the board or the interface software.
- The driver libraries are callable from most high level libraries.
- The driver system takes into account multiple boards (both opto-isolator and other supported boards) in the same computer.

Software interface to the PC-62B

Once the PC-62B has been installed in the computer and the external connections made, the board is in a state for obtaining status readings of the opto-isolated lines. This is done using I/O input commands. The PC-62B occupies two consecutive I/O addresses, starting at the board's base address as set by the base address DIP switch. When read, the first address reflects the status of the lines of Port A. The second address (offset 1 from the base address) reflects the status of Port B. If it is more convenient, the two addresses can be viewed as providing a single 16-bit value. The first address will contain the low order byte of this value (OIDATAL) and the second the high byte (OIDATAH).
Note that it is not necessary to initialise anything on the PC-62B in order to obtain readings of the status of the opto-isolated lines.

Reading the status of the Ports typically takes the form of the one following instructions:

- 'C'
```
status = inp( addr );
```
- Pascal
```
status = port[ addr ];
```
- Assembly Language
```
mov al, status
mov dx, addr
in al, dx
```

Where: `addr` is the I/O address of the opto-isolated port to be read to and `status` will contain the bit field read from the port.

Writing to the PC-62B's I/O addresses has no effect.

Software interface to the PC-65

Once the PC-65 has been installed in the computer and the external connections made, the board is in a state for controlling the opto-isolated lines. This is done using I/O output commands. The PC-65 occupies two consecutive I/O addresses, starting at the board's base address as set by the base address DIP switch. When written to, the first address controls the lines of Port A. The second address (offset 1 from the base address) controls the Port B lines. Writing a '1' to a bit in a port switches the corresponding line's output transistor on and writing a '0' switches it off. If it is more convenient, the two addresses can be viewed as a single 16-bit value. The first address will be the low order byte of this value (OIDATAL) and the second the high byte (OIDATAH).
Note that it is not necessary to initialise anything on the PC-65 in order to control the status of the opto-isolated lines.
Writing to the ports typically takes the form of the one following instructions:

- 'C'
```
inp( addr, status );
```
- Pascal
```
port[ addr ] = status;
```
- Assembly Language
```
mov al, status
mov dx, addr
out dx, al
```

Where: `addr` is the address of the opto-isolated port to be written to and `status` is the bit field to write to the output port.

Reading from the PC-65's I/O addresses produces meaningless results.

General Considerations

The opto-isolator board can be read from or written to at any time. There are no status bits to test or initialisation procedures to follow before communicating with the card.

5

Software

Full driver software is supplied as part of the opto-isolator package. It in provided in two equivalent forms:

- A driver software library.
- Complete source code for all driver routines.

The software library allows programmers to control the opto-isolator board via high level function calls, so allowing the user to write custom software applications without having to understand the low level operation of the board. Also included with the driver package is complete source code, in C and BASIC, for the entire driver package. This allows advanced users to modify existing code, rather than having to start writing low level code from scratch. The first part of this chapter gives details on using the library routines, and provides a full library function reference. The latter part of the chapter describes the source code, and gives suggestions on using it to tailor custom routines.

Using the Driver Software Library

The driver software library consists of a set of real-time functions for use with the opto-isolator board. The library routines are callable from most compiled languages, including the following:

- Microsoft C version 5.1
- Turbo C version 1.5
- Microsoft QuickBasic version 4.5
- LabWindows version 2.0

This is to say that the routines in the libraries as supplied are callable directly from programs written in the above languages. If the library does not operate correctly with your particular language compiler, then refer to the next section. It explains how to recompile the library source code with certain compilers.

Using the routines in the library is simple. In your program simply make references to the names of the routines, supplying appropriate parameters. Some languages need a header file included to provide information to the compiler about the library's contents. Do not forget to specify this in your program too. At program link time, along with your usual libraries, specify the name of the driver library you will be using so that the linker can link in to your program any routines from the library you have called.

Language Reference

'C': Microsoft C, Microsoft Quick C, Turbo C

Library name:	`DACQPCx.LIB`
Include file:	`DACQPC.H`

Notes: The *x* in the library name stands for the memory model of the corresponding library. *X* has the values:

s: Small memory model
m: Medium memory model
c: Compact memory model
l: Large memory model
h: Huge memory model

'LabWindows'

Library Name:	`DACQPC.LBW`
Include file:	`DACQPC.LWI`
Function panels:	`DACQPC.FP`

Notes: To install the driver for use with LabWindows, simply copy the three files to the LabWindows instrument directory. You can then use the instrument module as you would any other. The LabWindows code and functions are identical to the driver library system described in this chapter, but are supplied in the form of the three files mentioned above. These three files contain, in addition to the program code, instrument front panels and help information for use in the LabWindows integrated environment.

Return Codes

Every library function returns a code to indicate to the calling program the status of performing the function. The codes are:

`RETURN_OK`	Indicates that the function was performed correctly. Note that the name of the constant is `RETURN.OK` in BASIC. The numeric value is 0.
`ERR_NOT_AVAIL`	The `ERR_NOT_AVAIL` return code (`ERR.NOT.AVAIL` in Basic) indicates that the board was not available, or the requested function is not available for the board specified in the function call. The numeric value is -1.

ERR_PAR	A return value of ERR_PAR (ERR.PAR in BASIC) implies that at least one of the parameters passed to the function were out of range of not supported for the particular combination of board and function call. For example, specifying output mode to the DIO_prt_cfg function for the PC-62B would produce this return code. The numeric value is -2.

Multiple Boards

The driver library supports multiple boards in the same computer. The boards can be the same type or different types. All that is necessary to do to use the library with multiple boards is to call the Init_brd function once for each board installed in the computer. In doing this, the calling program assigns a board ID to the board being initialised. In all subsequent driver function calls, the board to which the function call is directed is specified by the board ID. Note that the different boards must have been configured to have non-overlapping base addresses and in general not use any of the computer's or other adapter board's resources that have already been used. For full details on all other boards currently supported please contact your distributor.

Library Function Reference

Quick Function Reference

The DACQPC driver as supplied with the opto-isolator board requires only six function calls. These are the following:

Init_brd	Initializes one of up to eight boards. This function initializes the hardware of the board if necessary and informs the driver system of its presence.
DIO_prt_cfg	Configures the operating mode of a port.
DIO_in_port	Returns digital data from the specified port.
DIO_out_port	Writes digital data to the specified port.
DIO_in_line	Returns the state of a particular line of the specified port.
DIO_out_line	Sets a particular line of the specified port.

Function Reference

Init_brd:

Name	`Init_brd`
Boards Supported	All
C Usage	`#include <DACQPC.H>` `int Init_brd( int iBrd_num, int iBrd_type, int iBase_addr );`
QuickBasic Usage	`REM $INCLUDE: 'DACQPC.INC'` `FUNCTION Init.brd%( BYVAL iBrd.num%, BYVAL iBrd.type%, BYVAL iBase.addr%)`
Description	This function initializes the board at the address `iBase_addr`. In subsequent calls to the driver system the board will be identified by the board number. The `iBrd_type` parameter must be set to the predefined constant `PC62BRD` for PC-62B boards or `PC65BRD` for PC-65 boards.
Return Value	`RETURN_OK` - Board initialized
Example	`DEMO62.C, DEMO65.C, DEMO62.BAS, DEMO65.BAS`

DIO_prt_cfg:

Name	`DIO_prt_cfg` - Configure a PC-65 port
Boards Supported	PC-62B, PC-65
C Usage	`#include <DACQPC.H>` `int DIO_prt_cfg( int iBrd_num, int iPort_num, int iLatch, int iDir );`
QuickBasic Usage	`REM $INCLUDE: 'DACQPC.INC'` `FUNCTION DIO.PRT.CFG%( BYVAL iPort.num%, BYVAL iLatch%, BYVAL iDir% )`
Description	This function configures the specified port for direction and handshaking mode. It is only necessary to call this function if the `DIO_out_line` function is to be used. The `iBrd_num` parameter is the number with which the board was initialized (via `Init_brd`). `iPort_num` is the number of the port to be configured. This ranges from 0 to 1. `iLatch` indicates the handshake mode. If this is 0, then the port is in normal (non-latched) mode. If this is 1, then latched mode operation is selected. This must be set to 0 for the PC-62B and PC-65 as the boards always operate in normal non-latched mode. `iDir` indicates the direction (input or output) that the port is configured for. 0 indicates input, and 1 output. For the PC-62B this parameter must always be 0 and for the PC-65 it must be 1.
Return Value	`RETURN_OK` - Port initialized. `ERR_NOT_AVAIL` - Port does not exist, or cannot be set to requested mode. `ERR_PAR` - One or more of the parameters were out of range or conflicting.
Example	`DEMO62.C, DEMO65.C, DEMO62.BAS, DEMO65.BAS`

DIO_in_port:

Name	`DIO_in_port` - Reads a byte from a PC-62B port.
Boards Supported	PC-62B
C Usage	`#include <DACQPC.H>` `int DIO_in_port( int iBrd_num, int iPort_num, int *iVal );`
QuickBasic Usage	`REM $INCLUDE: 'DACQPC.INC'` `FUNCTION DIO.in.port%( BYVAL iBrd.num%, BYVAL iPort.num, SEG iVal% )`
Description	This function reads digital input data from the specified port. The `iBrd_num` parameter is the number with which the board was initialized (via `Init_brd`). `iPort_num` is the number of the port from which to read. This can be 0 or 1. `iVal` is the result of the read operation. Note that `iVal` is passed by reference.
Return Value	`RETURN_OK` - Port read. `ERR_NOT_AVAIL` - Port does not exist.
Example	`DEMO62.C, DEMO65.C, DEMO62.BAS, DEMO65.BAS`

DIO_out_port:

Name	`DIO_out_port` - Writes a byte to a PC-65 port.
Boards Supported	PC-65
C Usage	`#include <DACQPC.H>` `int DIO_out_port( int iBrd_num, int iPort_num, int iVal );`
QuickBasic Usage	`REM $INCLUDE: 'DACQPC.INC'` `FUNCTION DIO.out.port%( BYVAL iBrd.num%, BYVAL iPort.num%, BYVAL iVal% )`
Description	This function writes a byte of digital output data to the specified port. Each bit set in the digital data byte will cause the corresponding output transistor of the port to be turned on. The `iBrd_num` parameter is the number with which the board was initialized (via `Init_brd`). `iPort_num` is the number of the port to be written to. This ranges from 0 to 1. `iVal` is the digital pattern to be written.
Return Value	`RETURN_OK` - Port written. `ERR_NOT_AVAIL` - Port does not exist.
Example	`DEMO62.C, DEMO65.C, DEMO62.BAS, DEMO65.BAS`

DIO_in_line:

Name	`DIO_in_line` - Reads the status of a single input line.
Boards Supported	PC-62B
C Usage	`#include <DACQPC.H>` `int DIO_in_line( int iBrd_num, int iPort_num, int iLine, int *iVal );`
QuickBasic Usage	`REM $INCLUDE: 'DACQPC.INC'` `FUNCTION DIO.in.line%( BYVAL iBrd.num%, BYVAL iPort.num%, BYVAL iLine%, SEG iVal% )`
Description	This function reads a single digital input line from the specified port. The `iBrd_num` parameter is the number with which the board was initialized (`via Init_brd`). `iPort_num` is the number of the port from which to read. This can be 0 or 1. `iLine` specifies the line to be read. 0 refers to the LSB, and 7 the MSB. `iVal` is the result of the operation. Note that `iVal` is passed by reference. The result is 0 for the line in the low state (input voltage less than 3V) and 1 for the line in the high state.
Return Value	`RETURN_OK` - Line read. `ERR_NOT_AVAIL` - Port does not exist.
Example	`DEMO62.C, DEMO65.C, DEMO62.BAS, DEMO65.BAS`

DIO_out_line:

Name	`DIO_out_line` - Writes a to a single isolated output line.
Boards Supported	PC-65
C Usage	`#include <DACQPC.H>` `int DIO_out_line( int iBrd_num, int iPort_num, int iLine, int iVal );`
QuickBasic Usage	`REM $INCLUDE: 'DACQPC.INC'` `FUNCTION DIO.out.line%( BYVAL iBrd.num%, BYVAL iPort.num%, BYVAL iLine%, BYVAL iVal% )`
Description	This function writes a single digital output line in the specified port. Other lines are not changed. Note that in order for this function to operate correctly, the `DIO_prt_cfg` function must have been called once for the specified port. The `iBrd_num` parameter is the number with which the board was initialized (via `Init_brd`). `iPort_num` is the number of the port in which the line is to be written. This ranges can be 0 or 1. `iLine` specifies the bit number of the line in the port to be written. 0 refers to the LSB, and 7 the MSB. `iVal` is the value to be written. It may be either 0 or 1. The value 0 switches off the output transistor corresponding to the bit number `iLine`, and 1 turns it on.
Return Value	`RETURN_OK` - Line read. `ERR_NOT_AVAIL` - Port does not exist.
Example	`DEMO62.C, DEMO65.C, DEMO62.BAS, DEMO65.BAS`

Source Code Modules

General

The source code of the driver software library is in the form of a single module. This module consists of a single program file, named DACQPC.C and a single include file, named DACQPC.H. Identical versions are supplied in C and in BASIC.

Note that if one is using routines from the supplied compiled libraries (detailed in the previous section) then there is no need to understand or use any of the information presented in this section. It is intended for those who need to write custom routines for their application. The source files thus may be modified to generate custom routines rather than having to write low level code from scratch. It would be useful to be familiar with the use of the various routines before modifying any of them for custom software.

Microsoft C, Turbo C and Microsoft QuickBasic compilers and the LabWindows system are specifically supported, but most other C or BASIC compilers should also be able to compile this module and its include file.

The driver system is designed to support multiple boards.

It is supplied with source code for both C and QuickBasic. These versions are identical, with the exception of the naming of procedures and constants. Where C names use an underscore ("_"), QuickBasic names make use of a period (".") For example, the initialization function is called `Init_brd` in C, and `Init.brd` in QuickBasic.

The port parameters to the routines are numbered from 0 to 1. Port 0 is Port A and port 1 is Port B.

The driver system uses the *return value error reporting method.* On exit, every routine returns a code indicating the result of performing the routine. Refer to the previous section for these return codes.

Use with specific languages

Microsoft C/Microsoft Quick C

All supplied modules are directly compatible with Microsoft C Version 5.1, as well as Quick C Version 2.01. You can compile the DACQPC.C module using any supported memory model. For example, using the large memory model:

```
cl /AL /c dacqpc.c
```

Required files	DACQPC.C, DACQPC.H
Examples	DEMO62.C, DEMO65.C

Turbo C

All supplied modules are directly compatible with Turbo C Version 1.5. You can compile the DACQPC.C module using any supported memory model.

Required files	DACQPC.C, DACQPC.H
Examples	DEMO62.C, DEMO65.C

LabWindows

The C and BASIC source files are directly compatible with the LabWindows integrated development environment.

Required files	DACQPC.C, DACQPC.H or DACQPC.BAS, DACQPC.INC
Examples	DEMO62.C, DEMO65.C, DEMO62.BAS, DEMO65.BAS

6

Hardware Specifications

External Interface

Number of input lines:	16 in two 8-bit ports
Input voltage (PC-62B):	0 - 3V is logic 0 3 - 25V is logic 1
Maximum reverse voltage:	PC-62B 400V peak
Input current	PC-62B: 1A peak 30mA continuous
Signal frequency response:	10kHz
I/O connector:	37 way female DB37 Amphenol or equivalent

Host computer interface

Bus type:	XT, AT, ISA, EISA
I/O address selection:	From 0h to 7F8h on 8 byte boundaries, DIP switch selectable.
I/O wait states:	0, 1, 2, 4 or 8 jumper selectable. Only affects cycles directed at the card.
Number of registers:	Two 8-bit
Word size:	8-bits
Power requirements:	+5V 1.0 mA typ, 100 mA max +12V 0.0 mA -12V 0.0 mA -5V 0.0 mA

Environmental

Operating temperature:	0°C to 55°C.
Storage temperature:	-55°C to 150°C.
Relative humidity:	5% to 90% non-condensing
Board size	PC-62B 16 x 10 cm PC-65 11.5 x 10 cm

Appendix A

Base address switch settings

Base Address	SW1	SW2	SW3	SW4	SW5	SW6	SW7	SW8
0h	ON	ON	ON	ON	ON	ON	ON	ON
8h	ON	ON	ON	ON	ON	ON	ON	OFF
10H	ON	ON	ON	ON	ON	ON	OFF	ON
18h	ON	ON	ON	ON	ON	ON	OFF	OFF
20h	ON	ON	ON	ON	ON	OFF	ON	ON
28h	ON	ON	ON	ON	ON	OFF	ON	OFF
30h	ON	ON	ON	ON	ON	OFF	OFF	ON
38h	ON	ON	ON	ON	ON	OFF	OFF	OFF
40h	ON	ON	ON	ON	OFF	ON	ON	ON
48h	ON	ON	ON	ON	OFF	ON	ON	OFF
50h	ON	ON	ON	ON	OFF	ON	OFF	ON
58h	ON	ON	ON	ON	OFF	ON	OFF	OFF
60h	ON	ON	ON	ON	OFF	OFF	ON	ON
68h	ON	ON	ON	ON	OFF	OFF	ON	OFF
70h	ON	ON	ON	ON	OFF	OFF	OFF	ON
78h	ON	ON	ON	ON	OFF	OFF	OFF	OFF
80h	ON	ON	ON	OFF	ON	ON	ON	ON
88h	ON	ON	ON	OFF	ON	ON	ON	OFF
90h	ON	ON	ON	OFF	ON	ON	OFF	ON
98h	ON	ON	ON	OFF	ON	ON	OFF	OFF
A0h	ON	ON	ON	OFF	ON	OFF	ON	ON
A8h	ON	ON	ON	OFF	ON	OFF	ON	OFF
B0h	ON	ON	ON	OFF	ON	OFF	OFF	ON
B8h	ON	ON	ON	OFF	ON	OFF	OFF	OFF
C0h	ON	ON	ON	OFF	OFF	ON	ON	ON
C8h	ON	ON	ON	OFF	OFF	ON	ON	OFF
D0h	ON	ON	ON	OFF	OFF	ON	OFF	ON
D8h	ON	ON	ON	OFF	OFF	ON	OFF	OFF
E0h	ON	ON	ON	OFF	OFF	OFF	ON	ON
E8h	ON	ON	ON	OFF	OFF	OFF	ON	OFF
F0h	ON	ON	ON	OFF	OFF	OFF	OFF	ON
F8h	ON	ON	ON	OFF	OFF	OFF	OFF	OFF
100h	ON	ON	OFF	ON	ON	ON	ON	ON
108h	ON	ON	OFF	ON	ON	ON	ON	OFF
110h	ON	ON	OFF	ON	ON	ON	OFF	ON
118h	ON	ON	OFF	ON	ON	ON	OFF	OFF
120h	ON	ON	OFF	ON	ON	OFF	ON	ON
128h	ON	ON	OFF	ON	ON	OFF	ON	OFF
130h	ON	ON	OFF	ON	ON	OFF	OFF	ON
138h	ON	ON	OFF	ON	ON	OFF	OFF	OFF
140h	ON	ON	OFF	ON	OFF	ON	ON	ON
148h	ON	ON	OFF	ON	OFF	ON	ON	OFF
150h	ON	ON	OFF	ON	OFF	ON	OFF	ON
158h	ON	ON	OFF	ON	OFF	ON	OFF	OFF
160h	ON	ON	OFF	ON	OFF	OFF	ON	ON
168h	ON	ON	OFF	ON	OFF	OFF	ON	OFF
170h	ON	ON	OFF	ON	OFF	OFF	OFF	ON
178h	ON	ON	OFF	ON	OFF	OFF	OFF	OFF
180h	ON	ON	OFF	OFF	ON	ON	ON	ON
188h	ON	ON	OFF	OFF	ON	ON	ON	OFF
190h	ON	ON	OFF	OFF	ON	ON	OFF	ON
198h	ON	ON	OFF	OFF	ON	ON	OFF	OFF
1A0h	ON	ON	OFF	OFF	ON	OFF	ON	ON
1A8h	ON	ON	OFF	OFF	ON	OFF	ON	OFF
1B0h	ON	ON	OFF	OFF	ON	OFF	OFF	ON
1B8h	ON	ON	OFF	OFF	ON	OFF	OFF	OFF
1C0h	ON	ON	OFF	OFF	OFF	ON	ON	ON
1C8h	ON	ON	OFF	OFF	OFF	ON	ON	OFF
1D0h	ON	ON	OFF	OFF	OFF	ON	OFF	ON
1D8h	ON	ON	OFF	OFF	OFF	ON	OFF	OFF
1E0h	ON	ON	OFF	OFF	OFF	OFF	ON	ON
1E8h	ON	ON	OFF	OFF	OFF	OFF	ON	OFF
1F0h	ON	ON	OFF	OFF	OFF	OFF	OFF	ON
1F8h	ON	ON	OFF	OFF	OFF	OFF	OFF	OFF

Table 2: Base address switch settings

Base Address	SW1	SW2	SW3	SW4	SW5	SW6	SW7	SW8
200h	ON	OFF	ON	ON	ON	ON	ON	ON
208h	ON	OFF	ON	ON	ON	ON	ON	OFF
210H	ON	OFF	ON	ON	ON	ON	OFF	ON
218h	ON	OFF	ON	ON	ON	ON	OFF	OFF
220h	ON	OFF	ON	ON	ON	OFF	ON	ON
228h	ON	OFF	ON	ON	ON	OFF	ON	OFF
230h	ON	OFF	ON	ON	ON	OFF	OFF	ON
238h	ON	OFF	ON	ON	ON	OFF	OFF	OFF
240h	ON	OFF	ON	ON	OFF	ON	ON	ON
248h	ON	OFF	ON	ON	OFF	ON	ON	OFF
250h	ON	OFF	ON	ON	OFF	ON	OFF	ON
258h	ON	OFF	ON	ON	OFF	ON	OFF	OFF
260h	ON	OFF	ON	ON	OFF	OFF	ON	ON
268h	ON	OFF	ON	ON	OFF	OFF	ON	OFF
270h	ON	OFF	ON	ON	OFF	OFF	OFF	ON
278h	ON	OFF	ON	ON	OFF	OFF	OFF	OFF
280h	ON	OFF	ON	OFF	ON	ON	ON	ON
288h	ON	OFF	ON	OFF	ON	ON	ON	OFF
290h	ON	OFF	ON	OFF	ON	ON	OFF	ON
298h	ON	OFF	ON	OFF	ON	ON	OFF	OFF
2A0h	ON	OFF	ON	OFF	ON	OFF	ON	ON
2A8h	ON	OFF	ON	OFF	ON	OFF	ON	OFF
2B0h	ON	OFF	ON	OFF	ON	OFF	OFF	ON
2B8h	ON	OFF	ON	OFF	ON	OFF	OFF	OFF
2C0h	ON	OFF	ON	OFF	OFF	ON	ON	ON
2C8h	ON	OFF	ON	OFF	OFF	ON	ON	OFF
2D0h	ON	OFF	ON	OFF	OFF	ON	OFF	ON
2D8h	ON	OFF	ON	OFF	OFF	ON	OFF	OFF
2E0h	ON	OFF	ON	OFF	OFF	OFF	ON	ON
2E8h	ON	OFF	ON	OFF	OFF	OFF	ON	OFF
2F0h	ON	OFF	ON	OFF	OFF	OFF	OFF	ON
2F8h	ON	OFF	ON	OFF	OFF	OFF	OFF	OFF
300h	ON	OFF	OFF	ON	ON	ON	ON	ON
308h	ON	OFF	OFF	ON	ON	ON	ON	OFF
310H	ON	OFF	OFF	ON	ON	ON	OFF	ON
318h	ON	OFF	OFF	ON	ON	ON	OFF	OFF
320h	ON	OFF	OFF	ON	ON	OFF	ON	ON
328h	ON	OFF	OFF	ON	ON	OFF	ON	OFF
330h	ON	OFF	OFF	ON	ON	OFF	OFF	ON
338h	ON	OFF	OFF	ON	ON	OFF	OFF	OFF
340h	ON	OFF	OFF	ON	OFF	ON	ON	ON
348h	ON	OFF	OFF	ON	OFF	ON	ON	OFF
350h	ON	OFF	OFF	ON	OFF	ON	OFF	ON
358h	ON	OFF	OFF	ON	OFF	ON	OFF	OFF
360h	ON	OFF	OFF	ON	OFF	OFF	ON	ON
368h	ON	OFF	OFF	ON	OFF	OFF	ON	OFF
370h	ON	OFF	OFF	ON	OFF	OFF	OFF	ON
378h	ON	OFF	OFF	ON	OFF	OFF	OFF	OFF
380h	ON	OFF	OFF	OFF	ON	ON	ON	ON
388h	ON	OFF	OFF	OFF	ON	ON	ON	OFF
390h	ON	OFF	OFF	OFF	ON	ON	OFF	ON
398h	ON	OFF	OFF	OFF	ON	ON	OFF	OFF
3A0h	ON	OFF	OFF	OFF	ON	OFF	ON	ON
3A8h	ON	OFF	OFF	OFF	ON	OFF	ON	OFF
3B0h	ON	OFF	OFF	OFF	ON	OFF	OFF	ON
3B8h	ON	OFF	OFF	OFF	ON	OFF	OFF	OFF
3C0h	ON	OFF	OFF	OFF	OFF	ON	ON	ON
3C8h	ON	OFF	OFF	OFF	OFF	ON	ON	OFF
3D0h	ON	OFF	OFF	OFF	OFF	ON	OFF	ON
3D8h	ON	OFF	OFF	OFF	OFF	ON	OFF	OFF
3E0h	ON	OFF	OFF	OFF	OFF	OFF	ON	ON
3E8h	ON	OFF	OFF	OFF	OFF	OFF	ON	OFF
3F0h	ON	OFF	OFF	OFF	OFF	OFF	OFF	ON
3F8h	ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF

Table 2 Base address switch settings (continued)