PC-35 prototyping board

User manual

INTERFACE PROJECTS FOR THE IBM PC USING THE PC-35 PROTOTYPING BOARD

CHAPTER 1
INTRODUCTION

IBM's Personal Computer (PC) was introduced in August, 1981, and was one of the most important landmarks in the short history of the personal computer industry. Its use of the 8088 CPU, its socket provision for the 8087 numeric coprocessor, and the wealth of software available, have all endeared the IBM PC to its users, but the decision to publish complete hardware documentation (available in the IBM Personal Computer Technical Reference Manual) has proved that it is genuinely a "personal" computer – one where the owner is given enough information to supplement the original design by interfacing his own circuits created for his own purposes. Whatever aspect of the "real world" one wishes to measure, the computer can be used as an extension of one's own senses: with its fast sampling ability and large storage capacity, it can be employed to detect subtle changes and relationships in the chosen field of study. The computer need not passively monitor – it can also be given the task of controlling temperature, pressure, light – anything for which suitable control interfaces and algorithms can be designed.

This manual introduces some of the basic concepts of IBM PC interfacing, using the PC-35 prototyping card. Why use a prototyping card? When building interfaces for the PC, more time is spent on the design of the circuitry that connects the board to the PC, than on the actual circuit of interest. The circuitry on the PC-35 is exactly this: the address decoder and data buffer (together with power supplies). The designer can now concentrate on prototyping the circuit he really wants, and get it going as quickly as possible.

The interface examples given are simple, but useful, but there are examples of software to control them. No hardware is of use unless the computer can be instructed to communicate with it. All the examples are well documented, but were written on the assumption that the reader has some knowledge of digital electronics and BASIC programming.

The manual begins with some definitions, and explanations of the concepts involved in interfacing to the PC. Next, to some of the simplest interfaces: the construction of input/output (I/O) ports. This is followed by the more complex subject of analogue to digital (A/D) and digital to analogue (D/A) converters, which require a considerable amount of theoretical discussion as to their characteristics (accuracy, speed, method of conversion) before any interface circuitry will make sense.

A digression from interfacing follows, in the form of a discussion of transducers. These are the devices that enable computers to interface to the real world: they

can convert a physical phenomenon (light level, temperature, etc) to an electrical signal which the computer can digitize and read.
In the control programs, comments and descriptive variables are used liberally. These will slow down the execution of the program, but speed up the user's understanding of it – it is easier to understand an instruction like OUT MUX ADDRESS, VALUE than the cryptic OUT X,Y. Programs can always be speeded up later.
The following standards are used in the hardware documentation:

* Logic signals that are active low are designated by capital letters followed by a hyphen (CS-). Active high signals are designated by capital letters only.
* Numbers with a trailing H (7F00H) are hexadecimal. Numbers sporting a trailing B (10011B) are binary, and numbers with no suffix must be assumed to be decimal.
* On a schematic diagram, a particular pin number of an IC (intergrated circuit) is prefixed by the U number of the IC (U6–3 is pin 3 of U6).
* A % refers to an integer variable – others are assumed real.

Devices which are too complicated and physically too large to be accommodated by the PC-35 are not discussed: nor are devices which are difficult to interface to the PC. The bibliography provides more reading for those who are interested in proceeding further.

CHAPTER 2
BEGINNING INTERFACING

Bus Buffering

Interfacing is the task of getting a computer to "talk" to an external device. The two major concepts involved are bus buffering and address decoding. Bus buffering is the connection of special circuitry to a particular bus, so that many external devices can be connected to these lines. One or two devices could easily be connected to the data bus of a system, but if, say, ten were connected, "bus loading" would occur. This is the inability of the computer to drive all the devices connected. Each device output is a load to the device driving it, and this can cause serious problems in a computer system. Only if the bus is buffered by a suitable bus buffer chip on the interface board can many devices be connected. The bus buffer is used to increase the driving power of a bus, and basically there are two types: drivers/receivers, and transceivers. A driver can source a relatively large amount of current so that it can drive a number of devices simultaneously, as would be the case when driving a bus. Receivers provide noise immunity (hysteresis), which reduces the chance of improperly detecting a data bit whilst listening to the bus. Devices such as the 74LS241 combine the properties of driver and receiver in the same gate. The transceiver, the remaining type of buffer, can pass information between two busses in either direction. When the data bus is buffered, transceivers are used, as the data must be both transmitted from and received by the CPU. Figure 2–1 illustrates the use of buffers with various system busses.

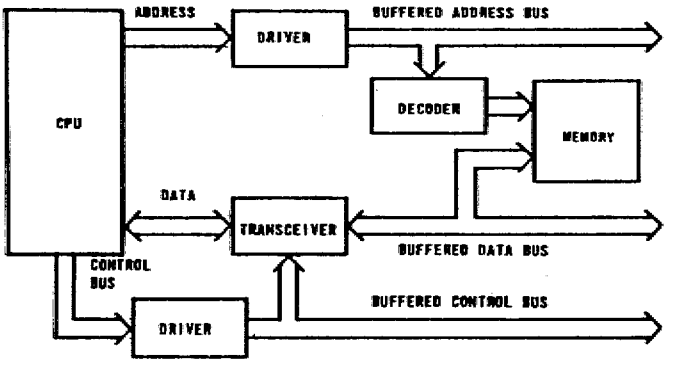


FIGURE 2–1. Block Diagram Showing the Use of Buffers

Address Decoding

For a computer to communicate with an external device, the device's address must be known. The computer will send this address out over the address bus, and then transfer the relevant data to or from the device. When the address is received at the address bus, it activates the address decoder, which is a special circuit used to monitor the address lines for an address or range of addresses. When that address appears on the bus, the address decoder produces a "device select pulse", which activates the device with which communication is intended. The device can then transmit or receive information to or from the computer, under control of the I/O reference commands (IN and OUT). Most address decoders are switch selectable: the user can programme the address which they will decode, by means of a switch.

Bus contention

It is important to avoid giving two devices the same address. The problem is called "bus contention", and can cause the computer to cease working entirely, or to behave in a seemingly irrational manner. Bus contention can be avoided by selecting an unoccupied location for the decoded address.
The design of the address decoder depends on what CPU is being used. There are two ways for a microprocessor to communicate with an I/O device: memory mapped I/O and I/O mapping. The first means that the microprocessor is treating the I/O device as a memory location, the second that it is treating it as an I/O port. I/O mapping is only possible with those CPUs which have a special I/O bus (8080, 8085, 8086, 8088). The 6502, 6800 and 68000 microprocessors do not have this I/O bus, and must consider all devices as memory locations. This requires more complex circuitry for implementing the address decoder, as well as using part of the memory space. With the IBM PC, the address decoder has only to decode 16 lines for the I/O bus (65536 locations), whereas for memory mapping 20 lines (1,048,576 locations) must be decoded.

I/O Mapping and Memory Mapping

The I/O address map, or memory map (if memory mapped I/O is to be used) should be consulted to see what addresses are available for interfacing use. These maps are charts showing the I/O and memory locations with descriptions of how they are used in the system. What addresses are available can be seen at a glance. Tables A1 and A2 give the I/O and memory maps of the IBM PC.

Software Commands

Software completes the interface. When a computer communicates with a memory location in BASIC, it uses the commands PEEK and POKE. When communicating with an I/O port, it uses the I/O reference instructions IN and OUT. While the IBM PC supports both sets of instructions, in these examples only the I/O instructions will be used. The commands will be more clearly explained in the chapter on simple I/O devices.

CHAPTER 3
THE IBM PC

System Configuration

Basically, the IBM Personal Computer consists of the System Unit and a Keyboard. The System Unit includes: resident cassette BASIC intepreter and operating system: cassette interface: resident diskette bootstrap loader: audio speaker: provision for up to 64K of on-board RAM: and I/O expansion capability. This last allows the installation of up to five option boards within the system unit. Options include 5.25" disc drive adaptors (two disc drives can be housed in the system unit), monochrome or colour/graphics display adaptor, game control adaptor, and many other special purpose options.
This is a basic description of the IBM PC. More detailed information can be found in "IBM Personal Computer Technical Manual" (IBM, 1981, Boca Raton, Florida), and "Inside the IBM PC" (Tenley Design: Starware, 1982, Washington, D.C.).

System Unit

The System Board is a large printed circuit board located within the chassis of the System Unit, along with a switching power supply. See fig. 3–1 for component location. Functionally, the board may be divided as follows: processor subsystem and support: Read Only Memory (ROM): R/W or Random Access Memory (RAM): I/O interfaces or adaptors, and the I/O channel.

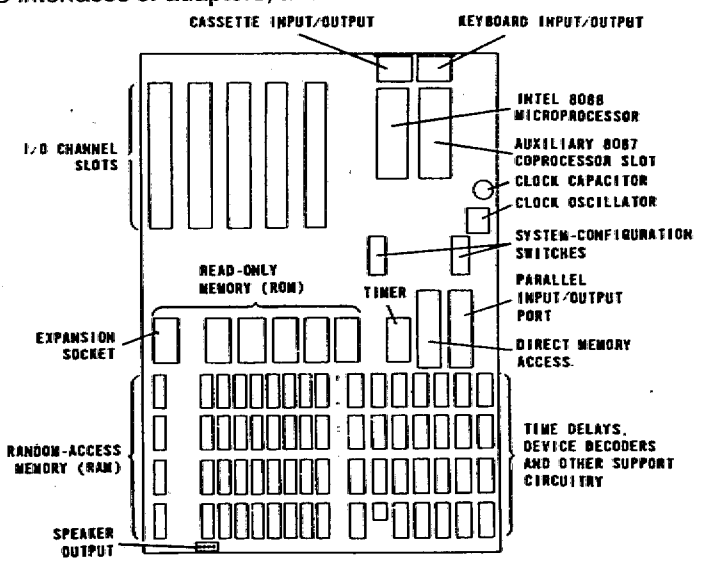


FIGURE 3–1. IBM PC Component Diagram

System Board

The IBM PC uses the Intel 8088 microprocessor, which is an 8-bit version of the 16-bit 8086 processor. The 8088 is software compatible with the 8086, and can be used with an optional 8087 coprocessor. In this mode, it provides high speed, 16-bit, floating-point multiply, divide, logarithmic, and trigonometric operations. 8087 and 8088 instruction may be used together, as the CPU will ignore 8087 instructions, and vice versa. As the 8088 supports 20-bit addresses, it is capable of up to 1 megabyte (MB) of storage in 16 kilobyte (KB) increments. The 8088 bus cycle consits of four 210nSec (4,77MHz) clock cycle, totalling 840nSec, but I/O and DMA cycles use five clock cycles (1,05μSec). Other features of this section are DMA capability, three timer/counters, and a prioritized interrupt controller.

The System Board's ROM or EPROM capability is 48K × 8, in six 8K × 8 partitions. Five are used for the cassette BASIC interpreter, cassette operating system, power-on self-test, I/O drivers, dot patterns for 128 graphics-mode characters, and a diskette bootstrap loader.

The RAM capacity is 64K × 9, 16K × 1 memory devices, with the ninth bit used for parity checking. About 4KB of RAM are used by the monitor in a cassette system. Memory cards are available which plug into the expansion slots, and provide a total of 1 MB of memory.

Table A1, the system memory map, shows ROM and RAM allocations.

The System Board includes interfaces/adaptors for an audio cassette, a serial keyboard, and a speaker. A cassette recorder can be connected to the PC via either the microphone or auxiliary inputs on the recorder. It can be started or stopped under program control. The keyboard interface provides for maximum flexibility in defining keyboard operations. It can request execution of a diagnostic that is performed by keyboard circuitry.

A 2.25" permanent magnet speaker is mounted inside the System Unit: its interface is capable of handling 0.5 watt of power, and allows for three modes of operation. One of the bits of the parallel I/O register may be toggled, and generate a pulse train: the timer/counter may be used to generate a waveform: the clock input to the timer/counter may be modulated with a parallel I/O register bit: or all three modes can be performed simultaneously.

The availability of the five edge card sockets on the System Board makes expanding the IBM PC's capabilities relatively simple. These are 62-pin connectors (TI part no. H421021–31) which are bussed together to extend the demultiplexed 8088 bus to the interface cards. Tables B1 and B2 give the connector pin assignments and signal descriptions.

The I/O CH RDY line can be used to interface slow I/O or memory devices to the PC. Processor generated read and write cycles use 840nSec (four CLK cycles) per byte transferred, but I/O read and write cycles and DMA transfers need 1.05μSec (five CLK cycles) per transferred byte. If the I/O CH RDY line is activated low true (NOT READY) the machine cycle is extended by an integral number of CLK cycles. An I/O CH CK–line is also available to report error conditions to the processor by initiating a Non-Mask Interrupt (NMI). Typically, this line will be used by memory for parity errors.

Four DMA channels are implemented in the PC system, but only channels 1 and 3 are available for use by devices installed in the expansion slots. Channel 0 is used for refresh of the dynamic RAM, and the disk controller will use channel 2.

Five of the eight available interrupt lines are reserved for particular system functions to be used by interface boards installed in the expansion slots, and one is used by the system timer channel. IRQ3 and IRQ5 are available for use by other devices in the expansion slots.
The System Board assumes two loads per expansion slot when buffering to provide drive to the interface cards. Although the system's I/O address space uses 16-bit port addresses, the ports on the system board are selected when address A9 = 0, reducing the I/O address map to 10K. The IBM PC expansion options listed in the IBM Personal Computer Technical Reference Manual all use a a 10-bit decode. Table A2, the I/O address map, shows the I/O resource allocations. Care must be taken when assigning addresses to other interface boards connected to the system, so as to avoid bus contention if any of the options are or will be used.

Power Supply

The IBM PC uses a switching power supply: 63.5 Watt, 120v AC, fused at 2 amps. It provides four regulated DC voltage levels and one 120v AC supply to power a display. See tables 3–1 and 3–2 for specifications of these voltages. The supply has overvoltage and overcurrent protection: should either of these conditions exist, the supply will shut down until it is rectified. The +5v DC output current is a maximum of 7 amps, of which 3 are used by the System Board. The other 4 amps are available for use by interface cards plugged into I/O channels. System dynamic RAM uses the +12v and −5v DC power supplies.

Table 3–1 – IBM PC Power Supply DC Voltage Output

Voltage (VDC)	Current (Amps)		Regulation	Tolerance
Nominal	Minimum	Maximum	%	%
+5	2.30	7.00	5	4
−5	0.00	0.30	10	8
+12	0.00	2.00	5	4
−12	0.00	0.25	10	9

Table 3–2 – IBM PC Power Supply Switched AC Voltage Output

Voltage (VAC)	Current (Amps)		Voltage Limits (VAC)	
Nominal	Minimum	Maximum	Minimum	Maximum
120	0.00	0.75	101.0	130.0

The power supply generates a "power good" signal which indicates the status of the +/−12v and +/−5v outputs. If these outputs are above the sense voltages listed in Table 3–3, the "power good" signal will be at a high level. Should one of the outputs fall below the sense voltages, the "power good" signal will be at a low level, which will trigger system shut-down.

Table 3–3 – ±5V and ±12V Sense Levels

	Sense Voltage (VDC)		
Output	Minimum	Nominal	Maximum
+5	+3.7	+4.0	+4.3
−5	−3.7	−4.0	−4.3
+12	+8.5	+9.6	+10.5
−12	−8.5	−9.6	−10.5

Keyboard

The keyboard is connected to the system unit by shielded 4-wire cable containing +5v DC (power), earth, and two bi-directional serial data lines. There is an Intel 8048 uC which performs the keyboard scan function, key debounce, and self test (which checks the memory, and checks for stuck keys). The processor can also buffer up to 20 key scan codes, maintain communications with the System Unit, and execute the handshaking sequence with each scan code transfer. Scan codes are used instead of ASCII (American National Standard Control Characters) codes so that the system software has flexibility in defining keyboard operations. Each key generates two codes – one which indicates that it is depressed, and one which indicates that it is not depressed.

Options

The I/O channel slots are used for optional devices which enhance the PC's utility. These options include: a monochrome display/parallel printer adaptor, used to interface to a display and an 80 CPS matrix printer: a colour/graphics monitor adaptor that can be used to control a colour monitor or television set, and includes a light pen input: a disc drive adaptor for one or two disc drives which can be housed in the System Unit: memory expansion in 32K × 9 and 64K × 9 configurations: a game control adaptor to connect joysticks, paddles, or switches to the system: and an asynchronous communications adaptor providing RS232C and current loop operation. Many other interface options are available from different manufacturers.

CHAPTER 4
THE PC-35 PROTOTYPING BOARD

The dimensions of the PC-35 prototyping card are 106mm high × 335mm long. It plugs into a system or expansion unit long slot. There are about 3,000 component points, 1mm (0.04") in size, with 1.5mm (0.06") pads. The component points are located on a 2,54mm (0.1") grid. All system control signals and voltage requirements are provided through a 2 × 31 position card-edge tab.
The PC-35 contains a voltage bus (+5v DC) and a ground bus (0v DC). Each bus borders the card, with the ground bus on the back (pin side) and the +5v bus on the front (component side). For greater design flexibility, a system interface giving eight chip-select lines and a bi-directional data bus buffer is also provided. The PC-35 has a 25-pin D-connector installed at the rear panel.
Important: The total width of the card, including components, should not exceed 12,7mm. Special sockets with short leads are available for wirewrap prototyping. If this specification is exceeded, components on the PC-35 may touch other cards plugged into adjacent slots.

Board Address Space

IBM specifications give the board address of the prototyping card as 300 – 31FH. Withing this address, eight chip-select lines are provided:

Chip-select line 0	300 – 303H
Chip-select line 1	304 – 307H
Chip-select line 2	308 – 30BH
Chip-select line 3	30C – 30FH
Chip-select line 4	310 – 313H
Chip-select line 5	314 – 317H
Chip-select line 6	318 – 31BH
Chip-select line 7	31C – 31FH

These chip-select lines allow easy interfacing of INTEL peripheral chips, such as 8255, 8254, 8253, etc., which occupy four successive port addresses. The chip-select lines can also be used to interface other digital circuits.

Bus Buffering

The following IBM bus lines are buffered, and can thus be loaded with up to 10LS TTL loads:

A0, A1
Reset
IOW, IOR
MEMW, MEMR
D0 – D7 (bi-directional)

Switching of the data buffer direction can be controlled by either IOW/IOR or MEMW/MEMR (selectable by jumbper JMP1), depending on the type of I/O data required.
All other IBM bus lines are accessible, and can be loaded with a maximum of two LS TTL Loads each.

System Loading and Power Limitations

Because there may be a large number of options interfaced to the system, the I/O bus loading should be limited to two LS TTL loads. This requirement will be met if the interface circuitry on the PC-35 is used.
The current requirements of circuitry in the prototyping area should not exceed the following limits:

+5v line	maximum 1.5A
+12v line	maximum 0.5A
-12v line	maximum 0.1A

CHAPTER 5
PROTOTYPE CONSTRUCTION

Before building up circuitry, it is necessary to consider component layout, installation, and interconnections.

Component Layout

Component locations should be assigned to minimize the length of interconnections. Components which use the data bus signals and OUT- and INPUT- pulses should be situated as close as possible to these connectors. Most circuits can be built from left to right, as shown in the diagrams.

Component Installation

Wire wrap or solder pin sockets can be used to install integrated circuits. In both cases, ensure that the total width of the card, including components, does not exceed 12.7mm, to avoid components on the PC-35 fouling adjacent cards.
The sockets should be securely positioned in the prototype area. Attach wirewrap sockets using either a socket with self-locking tapered terminals (Augat 314/316 series) or a silicon sealer such as RTV. Solder sockets can be soldered directly on to the pads.
Discrete components such as resistors, capacitors and transistors can be soldered directly to the solder pads in the prototype area.

Interconnections

Basically there are two methods of interconnecting prototype circuit components: either by soldering directly to the prototype area solder pads, or by wirewrapping to the square IC socket pins, or the component leads. As the component leads are usually round, they will not produce a good connection when wirewrapped unless they are soldered as well. Whatever method is adopted, it is as well to label the board with all IC numbers and pin numbers in the appropriate places. This will avoid confusion and speed up the proceedings.
Multistrand wire (22 gauge or smaller) is advisable when soldering to the prototype area pads, as single stranded wire breaks easily at a soldered connection. Components are held more securely if they are soldered to the pads as well as to their wirewrapping. Another method is to crimp multistranded wire to connectors which will slide on to the component or socket leads. Although this is an expensive and time-consuming process, it has the advantage that changes can be made quickly and easily, and that the wires may be reused in other circuits.
Wirewrapping is a popular and efficient technique developed by Bell Telephone Laboratories to cope with ever denser circuitry. Connections are made by coiling single stranded wire round the sharp corners of a terminal (fig. 5–1). The oxide

layer on both the wire and the terminal is crushed or sheared, forming a clean metal-to-metal contact (fig. 5–2). Wirewrapping wire and tools are commonly available.

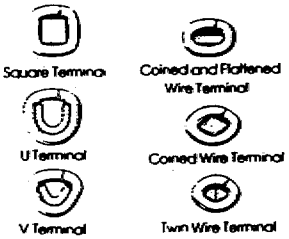


FIGURE 5–1. Terminals Suitable for Wire-wrapped Connections

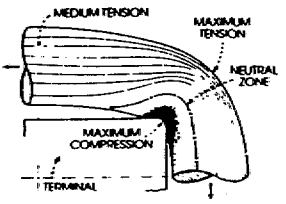


FIGURE 5–2. Cross-section of Wire-wrap Terminal Showing Contact Area

Figs. 5–3 to 5–5 show various wirewrapping problems. Fig. 5–3 is the result of not inserting the wire all the way into the wirewrapping tool. Pressing too hard will give overwrap (5–4), but constant pressure should be maintained, to avoid fig. 5–5. The first wrap on a terminal should be as close as possible to its base, and the last should not extend to the tapered or round part of a square post. Wirewrap pins should not be bent or twisted.

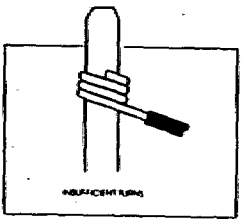


FIGURE 5-3. Incorrect Wire-wrap Connection: Insufficient Turns

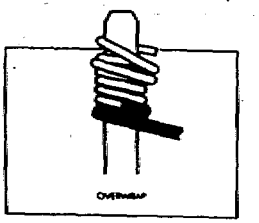


FIGURE 5-4. Incorrect Wire-wrap Connection: Overwrap

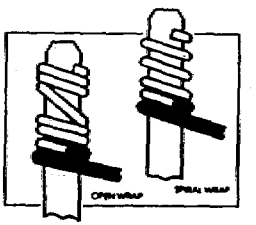


FIGURE 5-5. Incorrect Wire-wrap Connection: Open Wrap and Spiral Wrap

An important consideration with prototyping circuitry is wire routing. Usually it is easiest to begin with the various power supplies, and a "daisy chain" connection can be used (5-6).

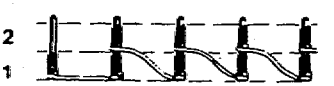


FIGURE 5-6. "Daisy Chain" Connection

Orient the wire coming from a terminal to avoid unwrapping connections. Interconnecting runs should be as random as possible, as crosstalk tends to occur with parallel runs. Interconnecting wire should not be taut, to avoid putting stress on the wires and terminals. Wire buildup in any part of the circuit should be avoided.

CHAPTER 6
SIMPLE I/O DEVICES

Input and output parallel ports are the simplest and easiest I/O devices to connect to the PC bus. They enable the computer to communicate with anything external to itself, as a computer port is where data enters and leaves the machine. Parallel I/O ports transfer several bits of data simultaneously, but serial ports transfer data one bit at a time, over a single line. Serial I/O techniques are not dealt with here, but the bibliography contains references to works which can be consulted.
An input port enables the computer to read information present at its input, which could come, for example, from another device such as an analogue to digital converter (ADC), or as the representation of the status of a device. Output ports transfer data from the computer to the outside world, to be used by such devices as digital to analogue converters (DAC) and printers. In the IBM PC, 65,536 I/O port locations are available.

I/O Software Commands

It is of course important to be aware of the software commands which are used to transfer data to and from the computer. The IBM PC uses the iAPX8088 microprocessor as its CPU (central processing unit). The CPU can communicate with a device in two ways: treating the device as a memory location and using the memory reference instructions PEEK and POKE to transfer data (memory mapped I/O): or treating the device as an I/O port, and using the I/O reference instructions OUT and INP (I/O mapping). This latter method will be considered here. Use of I/O mapping makes the construction of the address decoder simpler, and reduces the chances of bus contention. Table 6–1 shows the general format of PEEK, POKE, INP and OUT commands.

Table 6–1

Basica Memory Reference Instructions		
INPUT:	X = PEEK (n)	(n range: 0 to 65535)
OUTPUT: (m range: 0 to 255)	POKE n,m	(n range: 0 to 65535)
Basica I/O Reference Instructions		
INPUT:	X = INP(n)	(n range: 0 to 65535)
OUTPUT: (m range: 0 to 255)	OUT n,m	(n range: 0 to 65535)

In the PC interpreter are a number of I/O commands, including LPRINT, PRINT, INPUT, SAVE and LOAD. Such commands are not generally thought of as I/O commands, as they communicate with standard peripherals (CRT, printer, disk drives). However, they are device-specific I/O commands.

INP and OUT cause a definite sequence of events to occur when they are executed. During execution, each command's address information is placed on the address bus. The OUT command contains a data byte that is placed on the data bus: INP reads the data placed on the data bus by the I/O device being addressed.

Output Ports

Output ports transmit data from the computer to the outside world, and are easily constructed from D-flip flops, such as the 74LS373, 74175, and 7475 standard TTL series of ICs. D-flip flops latch the transient data from the data bus, retaining the information until it is written over. Output ports are used to transfer data to devices such as DACs, floppy discs, and printers, as well as their use in control applications. They can be used to activate or control external devices (LEDs, pumps, fans, relays, etc) by setting certain of their bits.
To communicate with an output port that is configured as an I/O device at location 300H, the command

OUT &H300,0

would cause the pattern 00000000 (binary version of decimal 0) to appear at the output port. Command

OUT &H300,255

would cause the pattern 11111111 (binary version of 255) to appear.

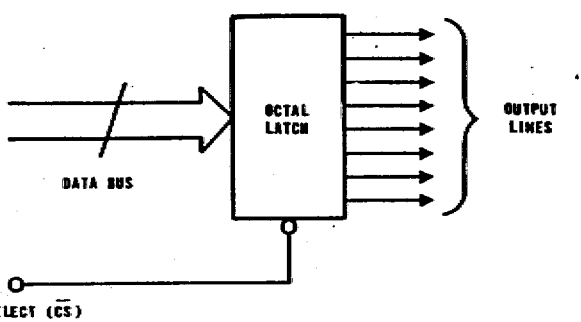


FIGURE 6-1. Output Port Block Diagram

Input Ports

Input ports receive data, usually 8 bits at a time, from the outside world, for transmission to the computer. They can be constructed from bus buffer chips such as the 74125 and 74LS244, which have tri-state outputs that electrically disconnect them from the data bus, so that they are only connected when addressed by the computer. The interface is thus simplified by the prevention of problems with bus loading.

Typically, the transmission of information to the computer via an input port takes place 8 bits at a time. The information may represent data, or the state of external devices, and could be an A/D converter or card reader data value, or the input from a sensor connected to one input line, which changes state (0 to 1) when a certain threshold is reached. The input port bit pattern would reveal the state of the sensor.

If an input port were configured at location 300H, the command

A = INP (&H300)

would assign the bit pattern on the input of the port to the variable A. A bit pattern of 00000000 would make the value of variable A 0, and one of 11111111 would make its value 255.

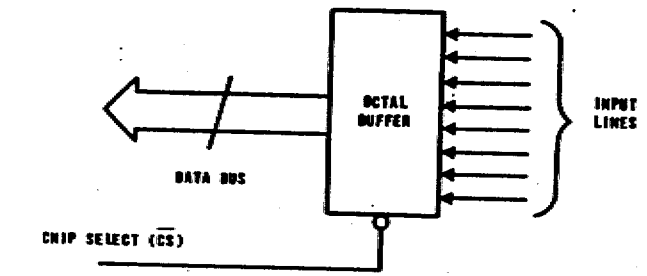


FIGURE 6–2. Input Port Block Diagram

Optoisolation

This is used to protect the computer from electrical damage when controlling or monitoring an external device, and is the electrical isolation of the computer from the outside world. Electrical inputs or outputs are first converted to light beams, using a LED, and later reconverted to electrical signals, using a phototransistor. The optoisolator contains both the LED and phototransistor. Thus the only connection between the computer and the outside world is a beam of light.

Optoisolators are usually connected at the input of an input port, and at the output of an output port.

CHAPTER 7
REAL WORLD INTERFACING – INPUT & OUTPUT

Real world interfacing is the connection of the CPU to external devices, to enhance its capabilities. The following circuits can all be used with a simple input or output port. Operation of the circuits is by a change of state when a preset level is reached.

LEDs and Lamps

A LED (light emitting diode) is one of the easiest devices to control from an output port. It is as simple as connecting the LED in series with a 220 ohm resistor (to limit LED current) to the output of an open collector gate. The LED will switch on when the input of the inverter is high.

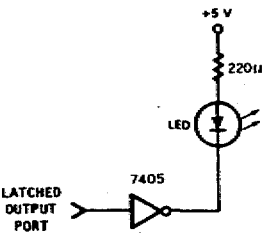


FIGURE 7-1. Interfacing to an LED

A similar circuit uses a small lamp instead of a LED. A transistor is needed to handle the higher current needed by an incandescent lamp.

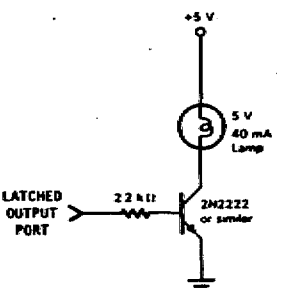


FIGURE 7-2. Interfacing to a Lamp

Seven-segment displays, commonly used in computer systems to display data, can easily be connected to an output port via 330 ohm resistors.

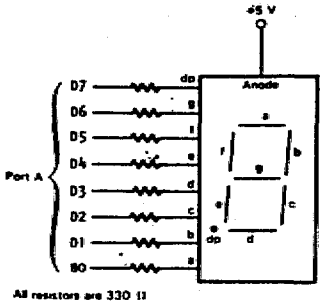


FIGURE 7–3. Connecting 7–segment Display Devices to Output Ports

Relays

Relays can be interfaced to output ports via open collector gates. It may be necessary to limit the current through the relay coil by connecting a series resistor to the output of the gate. The diode in parallel with the coil absorbs the reverse current created by the magnetic field caused when the coil is de-energized. Some relay packages include this diode.

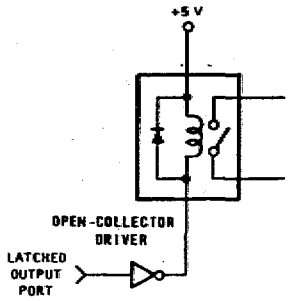


FIGURE 7–4. Controlling a Relay

Switch Inputs

Reading the open or closed condition of a switch is the most fundamental computer input operation. A simple circuit that can be connected to the line of an input port is shown in fig. 7–5. When the switch is open, the input line is connected to +5v, and is thus high or logic 1. When the switch is closed, the line is earthed (logic 0). The line can be tested for high or low state by using an INP command and the logical operator AND to mask off unwanted bits.

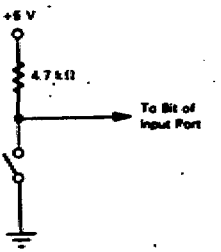


FIGURE 7–5. Connecting a Switch to an Input Port

Light

This is commonly monitored, in solar energy, assembly line counting, remote control, and so forth. Figs. 7–6 and 7–7 show the use of a photocell and phototransistor connected to one bit of an input port. Both circuits change state (0 to 1) when a fixed light intensity has been reached, there being no adjustment for the light level. 7–8 shows a circuit with a sensitivity adjustment, which can be electrically set to change state at a desired light level.

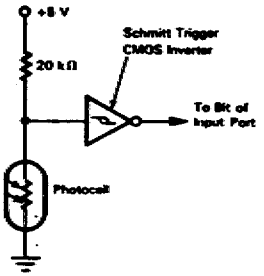


FIGURE 7–6. Light Detection Using a Photocell

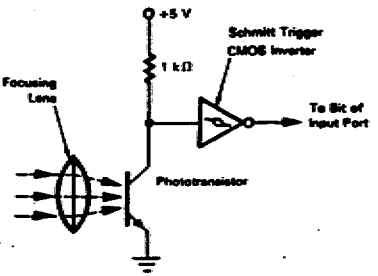


FIGURE 7–7. Light Detection Using a Phototransistor

Like the switch inputs, these circuits can be used as a single bit port input by using the INP command. Several circuits can be connected to different inputs on a single input port. Logical operator AND can be used to mask out unwanted bit positions.

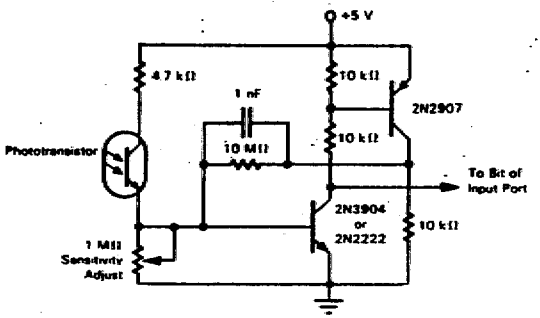


FIGURE 7–8. Light Detection Circuit With Sensitivity Adjustment

Temperature

The active element of the simple heat sensor in fig. 7–9 is a thermistor. Ambient temperature change changes the resistance of the thermistor, and, as the temperature increases, the output changes state from logic 1 to logic 0. As the temperature setpoint cannot be adjusted, it is only useful for simple applications.

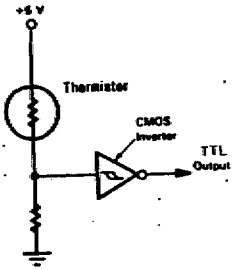


FIGURE 7–9. Thermal or Heat Detector

Precision temperature sensing with adjustable trip points is shown in fig. 7–10. This circuit is based on the National Semiconductor LM3911 temperature sensing chip, and it operates from +12 and +5 volts. Its applications include energy management, solar energy, and environmental control.

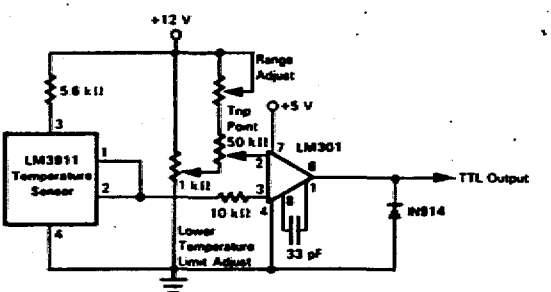


FIGURE 7–10. Temperature Sensing Circuit With Adjustable Trip Point

Conductivity

Conductivity is the basis for sensors monitoring flooding and the presence of rain. A simple conductivity sensing circuit is shown in fig. 7–11. It consists of an interconnected grid pattern on a printed circuit board. When the sensor is dry, there is no contact between the lines: when they are wet, the water shorts them, and the output changes form a high to a low logic level at a particular conductivity level.

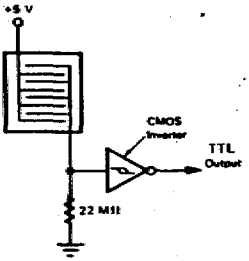


FIGURE 7–11. Conductivity Sensing Circuit

Touch

Fig. 7–12 shows a simple touch plate circuit, useful for security systems. It is based on the NE555 timer, used in the monostable multivibrator mode.

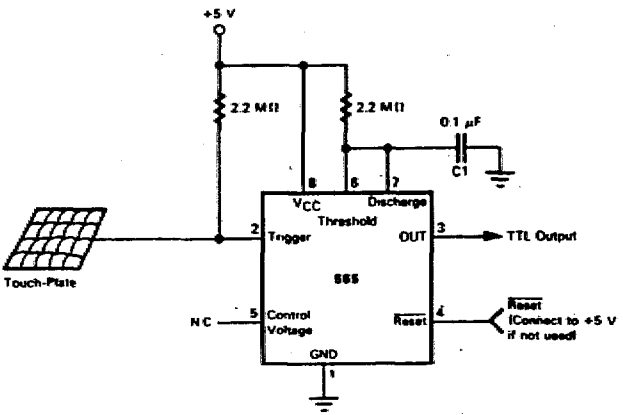


FIGURE 7–12. Touch–plate Sensing Circuit

If the touch plate is touched, the trigger input (pin 2) pickes up the 50Hz field present in almost all buildings, and generates an output. This circuit is not suitable for all applications, as it is susceptible to noise.

Optoisolation

An excellent method of protecting the CPU from high or transient voltages is to use optoisolation (chapter 6). A representative circuit is shown in fig. 7–13.

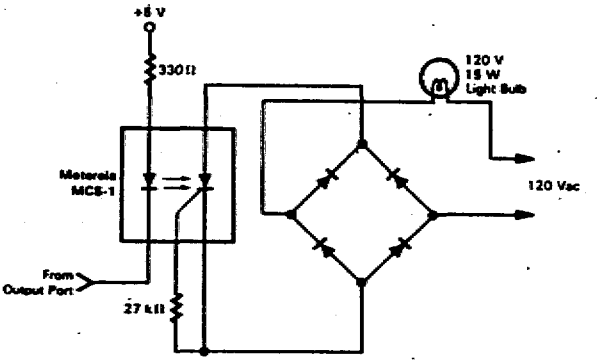


FIGURE 7–13. AC Control Using an Opto-coupler/SCR

Voltage detection

The circuits in figs. 7–14 to 7–16 are used to monitor the presence of voltages. Connection of the voltage direct to the input line could damage the computer system, so optoisolation is used.

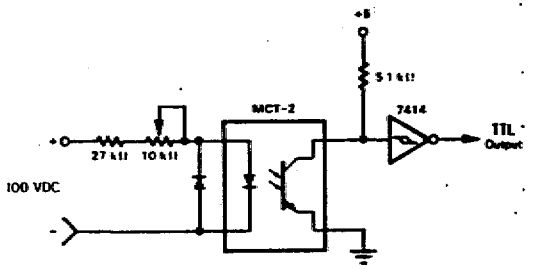


FIGURE 7-14. DC Voltage Detection Circuit

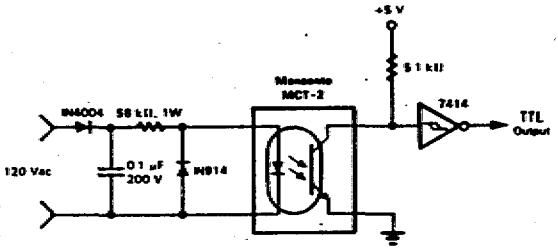


FIGURE 7-15. Line Voltage Detection Circuit

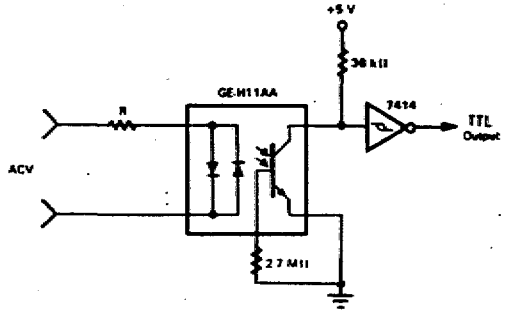


FIGURE 7-16. Basic AC Voltage Sensor Circuit Using GE H11AA

CHAPTER 8
ANALOGUE INTERFACING

Real world applications of microcomputers often require the measurement of levels which cannot be expressed in the TTL levels of 0v and +5v existing in the computer. The computer might have to measure the output voltage of a temperature transducer, or generate an analogue voltage to drive a chart recorder. Situations exist where analogue I/O capability is required to interact with the analogue real word outside the computer's binary world. To measure an analogue signal, an analogue to digital converter (A/D or ADC) is used, and to generate an analogue voltage a digital to analogue converter (D/A or DAC).

Digital to Analogue Converters

A DAC can be tought of as a digitally programmable resistor network, accepting digital signals at its input, which generates a voltage or current output representative of the digital input. When using a black box representation of a DAC, the only concern is with the control and data lines entering or leaving the device, and not with its actual action.

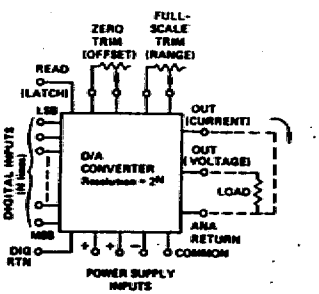


FIGURE 8-1. "Black Box" Representation of a Typical DAC

The theoretical transfer function refers to the type of analogue output that can be expected for a given digital input. It is not smooth, but consists of a series of steps (discrete voltages).

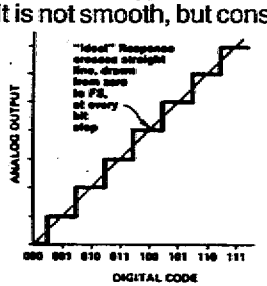


FIGURE 8-2. Theoretical Transfer Function of a DAC

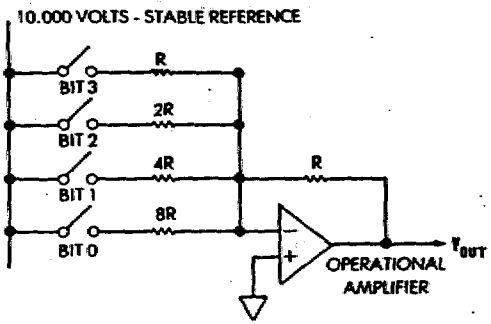


FIGURE 8-3. A Simple 4-bit D/A Converter

Fig. 8–3 shows a resistor network which generates an analogue output (Vout) related to the value of the input digital signal (D) and the voltage reference by the equation:

$$Vout = D * Vref$$

where Vout = voltage output of the DAC
D= value of the digital input
Vref= DAC voltage reference.

Analogue I/O interfaces which can represent both negative and positive voltages are called bipolar converters. Those which handle only positive or negative voltages are called unipolar converters.
The resolution of a DAC is the smallest possible step change in output voltage, and is related to the number of input bits. It is usually given by the equation:

$$\text{Voltage Resolution} = (1/2^{**}N) * (\text{F.S.})$$

where N= Number of bits
F.S.= Full Scale output.

The resolution of a DAC, expressed as a percentage, is given by the equation:

$$\text{Resolution (\%)} = (1/2N^{**}) * 100$$

Expressed as a percentage, voltage resolution for various numbers of bits is shown in table 8–1.

Table 8–1 – DAC Voltage Resolution Expressed as a Percentage

NUMBER OF BITS	2**N	RESOLUTION (%)
1	2	50.00
2	4	25.00
3	8	12.50
4	16	6.250
5	32	3.1250
6	64	1.5625
7	128	0.78125
8	256	0.391625
9	512	0.195313
10	1024	0.097656
11	2048	0.048828
12	4096	0.024414
13	8192	0.012207
14	16384	0.006104
15	32768	0.003052
16	65536	0.001526

The maximum voltage output for a unipolar converter is given by the equation:

$$Vmax = Vfs\ (1 - (1/2^{**}N))$$

where Vmax = maximum voltage output
Vfs = full-scale range
N = number of data bits input to the converter.

E.g., a 12-bit unipolar device with a +10v full-scale range would have the following specifications:

Vfs = 10.0000 volts
Vmax = 9.9976 volts
Vmin = 0.0000 volts

and its resolution would be (1/4096) * 10.00v, or about 2.44mV.

Analogue to Digital Converters

Analogue to digital converters (ADCs or A/Ds) are used to convert a continuous analogue input into its quantized binary equivalent.

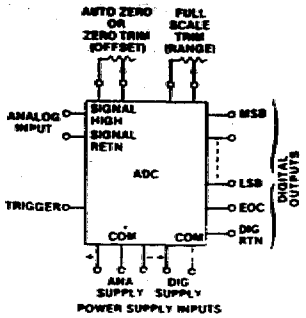


FIGURE 8–4. "Black Box" Representation of an A/D Converter

Many methods of analogue to digital conversion are popular, but only the two which dominate the market at present will be discussed here: successive approximation, and integrating A/D conversion. If a rapidly changing signal must be frequently sampled to prevent information about it being lost, the faster successive approximation converter is the best choice. 15 or 25uSec conversion, with 8- or 10-bit resolution, can easily be obtained at reasonable cost. These ADCs are used when the application demands moderate speed and resolution. In commercially available units, the ability to convert an analogue signal to 8 bits resolution in 1uSec, and 16 bits in 50uSec, is common.
Most converters can be triggered by using a START CONVERSION (START) signal. All have a BUSY or END OF CONVERSION (EOC) line to indicate when a conversion is in progress.

Successive Approximation A/D converter

Although the diagram (fig. 8–5) shows 8-bit resolution, the technique involved can be extended to cover any resolution. The three major components are:

1. Comparator
2. Successive approximation register (SAR)
3. Digital to analogue converter.

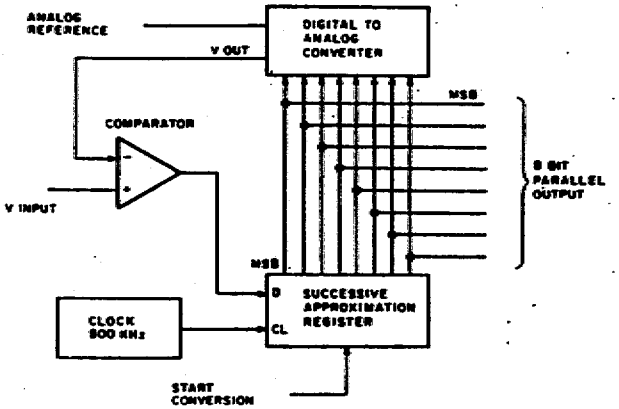


FIGURE 8-5. Block Diagram of Successive Approximation A/D Converter

Successive approximation converters operate by successively comparing the analogue input signal to fractions of a reference signal. As the comparator circuitry makes each comparison, the output of the comparator indicates whether the input signal is greater or less than the D/A converter output signal, and sets or resets the bit for which the comparison is being made. This continues with each bit, until conversion is complete.

Newer ADCs can be represented by the Intech DAS5716, a 16-bit, 16-channel data acquisition system (DAS) on one chip. It includes an analogue input multiplexer (MUX), an adjustable gain instrumentation amplifier and a sample and hold amplifier, as well as the actual A/D converter, and its complete system conversion time is 70uSec. Interfacing is greatly simplified by treating the DAS as 16 sequential memory locations.

A rapidly changing signal can induce error by changing value during the conversion cycle. This can be prevented by using a sample and hold amplifier (SHA) between the analogue input and the A/D converter. Some systems, such as the DAS5716, incorporate a sample and hold amplifier as standard.

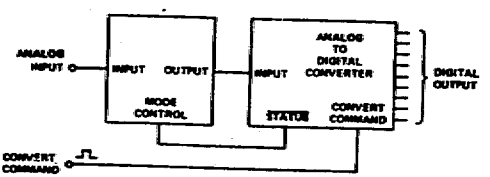


FIGURE 8-6. Connecting a S/H Amplifier to an A/D Converter

If a periodic analogue signal is being sampled, it must be done often enough to enable an adequate reconstruction. According to sampling theory (Shannon's theorem) the signal must be sampled at a rate which is twice the frequency of the highest frequency component, in order to retain the ability to reproduce the signal. If a signal is sampled too slowly, the problem of aliasing, the impersonation of a lower frequency signal by one of a higher frequency, can arise.

Integrating A/D Converter

If a fast A/D converter is used in noisy environments, there is the danger of digitizing a noise spike, which means that the value of the input signal can never be certainly known unless a filtering algorithm is being used in the program. An integrating A/D converter allows a capacitor integrator to be charged by the input voltage, and measures the time taken to do this. Using a reference voltage for comparison, a value for the input voltage can be determined. Integrating A/D converters tend to average the signal and so are less susceptible to noise. They are, however, very slow, taking as long as 100mSec for one conversion. Intersil's ICL7109 is a representative device.

CHAPTER 9
ANALOGUE SIGNAL CONDITIONING

Transducers are used to convert such physical phenomena as temperature and pressure into electrical signals. Signal conditioning circuitry amplifies, scales, and otherwise converts the output of the transducer into a voltage compatible with the input of an A/D converter. Signal conditioning circuitry usually contains operational amplifiers or transistorized circuitry with gain and offset adjustments.

Temperature

This is probably the most commonly monitored external influence. Among the many temperature measuring transducers are themocouples, resistance temperature detectors (RTDs), thermistors, and solid state sensors. There is space here for only a few circuits to illustrate general principles.

Thermistors

The basic thermistor circuit consists of a thermistor in series with a resistor. The voltage drop across the thermistor is a function of temperature. It is a simple circuit to use, but non-linear, and the output must be calibrated to a non-linear equation.

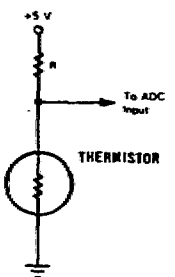


FIGURE 9-1. Basic Thermistor Circuit

Solid State Sensors

The National Semiconductor LM3911 is an accurate temperature measuring chip for use over the range −25 to +85 degree C. The voltage output is 10mV per degree Kelvin, linear over the operating range. This chip is available in several packages, including a 4-lead TO-5, and an 8-pin mini-DIP. The diagram shows how the chip can be connected to give a signal output with offset adjustment.

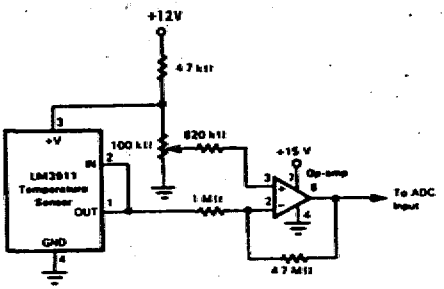


FIGURE 9–2. LM3911 Temperature Sensor Circuit

Another solid state temperature sensor is the Analog Devices AD590. This device has only two leads, and is available in a stainless steel tubular probe (AC2626), for liquid and gaseous immersion applications. A simple but accurate thermometer circuit is shown in fig. 9–3. In fig. 9–4, the use of two AD590s for differential temperature measurement is shown. A possible use of this circuit could be to measure the temperature difference between a black object and a white or reflecting object, to obtain an accurate measurement of the amount of sunlight.

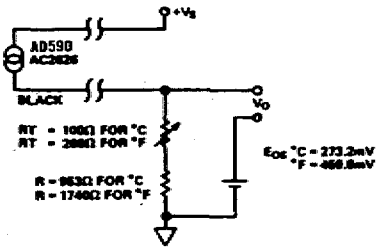


FIGURE 9–3. Simple Thermometer Using AD590

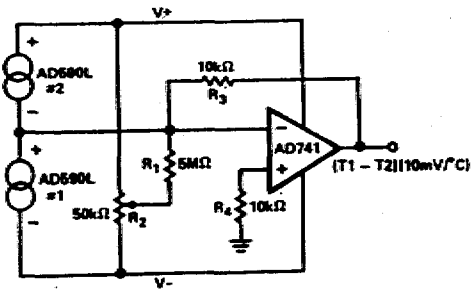


FIGURE 9–4. Differential Temperature Measurement

Solid state temperature sensors are useful for several reasons: linearity, ease of interfacing, and, not least, cheapness. They are also constant current devices, which means they can be used some distance from the circuit without loss of accuracy. They can be used to monitor temperature around a whole building, and still maintain accuracy.

Light

Light can be measured using photocells and phototransistors as transducers. The basic photocell circuit resembles the basic thermistor circuit. Fig. 9–6 shows an improved circuit with a phototransistor to improve sensitivity, and fig. 9–7 includes sensitivity adjustment.

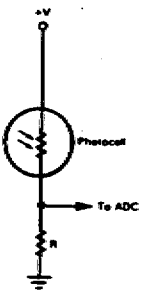


FIGURE 9–5. Basic Photocell Circuit

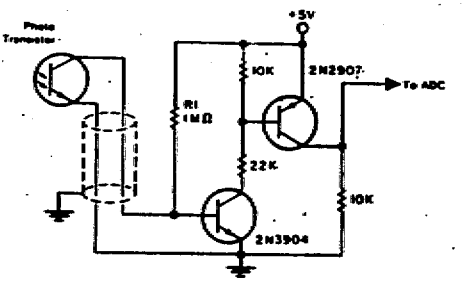


FIGURE 9–6. Photo-transistor Circuit

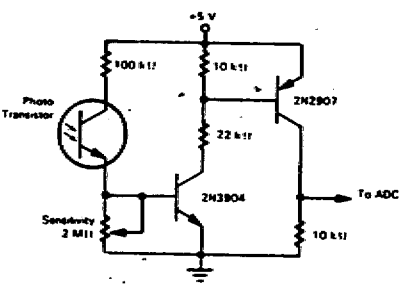


FIGURE 9–7. Photo-transistor Circuit With Sensitivity Adjustment

Overvoltage protection

It is good practice to include overvoltage protection to circuit designs. The circuit in fig. 9–8 is simple but effective, using two zener diodes back-to-back in parallel to protect the circuit from positive and negative overvoltage. The zeners will clamp the input voltage and not let it rise above their value. Zener value should be above the maximum voltage expected, but below the maximum voltage rating of the analogue input circuit.

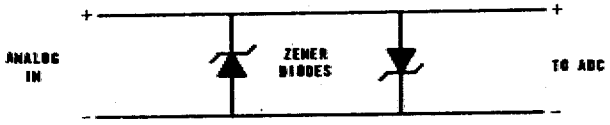


FIGURE 9–8. Overvoltage Protection Using 2 Zener Diodes

CHAPTER 10
CERTAIN EXPERIMENTS

This chapter gives examples of interfacing projects using the PC-35 prototyping board, as well as the software required to control the interface. These experiments can be considered the elements of more advanced designs.

First Experiment: 8–Bit Output Port

Parts required:

1 20-pin wirewrap IC socket
1 74LS374 octal latch

This experiment details the construction of an 8-bit output port, which can be used, under computer control, to route data from the computer to an external peripheral device which will hold or latch the data until it is written over. The 74LS374 is the 8-bit latch.
Construct the circuit shown in fig. 10–1 on the PC-35 board. Note: under no circumstances should the PC-35 board be plugged into the computer with power on, as this can damage both the PC and the PC-35. Connections and disconnection of all kinds should be made with power off.

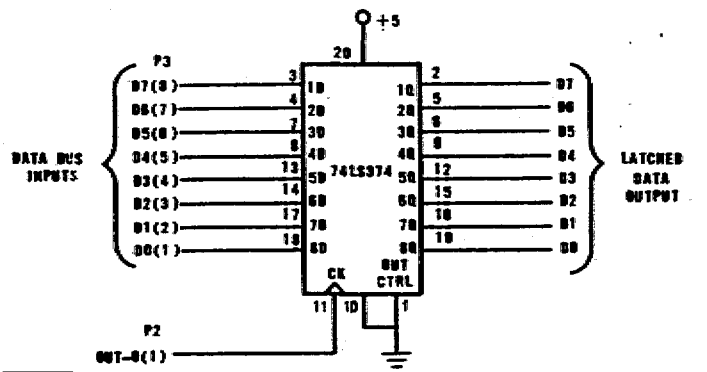


FIGURE 10–1. 8–bit Output Port

The output port can be tested after construction by using the OUT command. This takes the general form:

OUT location, value

where location = I/O location to transfer data. Valid locations are 300H to 31FH
value= Data to transfer to the I/O location. Valid range is 0 to 255

Use chip select line 0, connect a voltmeter to the latch outputs, and execute the command:

```
OUT &H300,255
```

This will send the decimal value 255 (binary 11111111) to the latch, causing all outputs to be logic 1. Now execute the command:

```
OUT &H300,0
```

which will cause all output bits to be logic 0.

A visual LED indicator can be constructed as in fig. 10–2. The LED will light whenever its corresponding bit is logic 1.

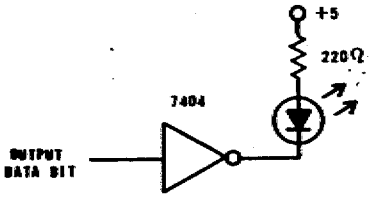


FIGURE 10–2: Connecting an LED to the Output Port

This program turns the LEDs on one at a time:

```
100  REM BINARY COUNTING PROGRAM
110  REM
120  CLS:KEY OFF
130  ADDR = &H300
140  FOR I = 0 TO 7
150       VALUE = 2^1
160       OUT ADDR, VALUE
170  REM
180  REM      JUST WASTING TIME . . .
190  REM
200       FOR J = 1 T0 1000
210       NEXT J
220  NEXT I
230  END
```

Second Experiment: 8-bit input port

Parts required:

1 74LS244 octal buffer
1 20-pin IC wire-wrap socket

The 74LS244 octal buffer has tri-state outputs, and is used in this experiment as an input port, used to channel data from the outside world to the computer. Use the diagram (fig. 10–3) to construct the circuit on the PC-35 board.

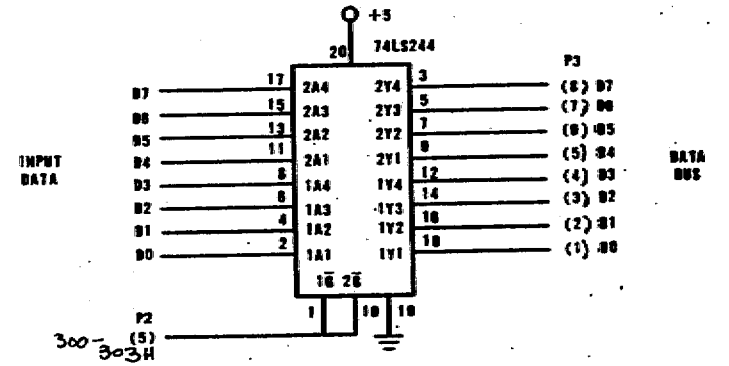


FIGURE 10–3. 8–Bit Input Port

Read in data to check the board. Simulate input data by connecting the input lines as shown in fig. 10–4, which will maintain the input line at logic 1.

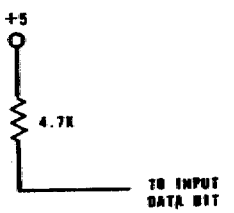


FIGURE 10–4. Connecting +5V to an Input Port

To read the input of the port, using chip select line 0, enter the command:

A = INP (&H300)

The variable A will now contain the decimal value of the bit pattern present on the input port.
Fig. 10–5 shows the construction of a 16-bit output port from two 8-bit ports. Two INP commands would read 8 bits at a time, and combine the two bytes into a 16-bit software word.

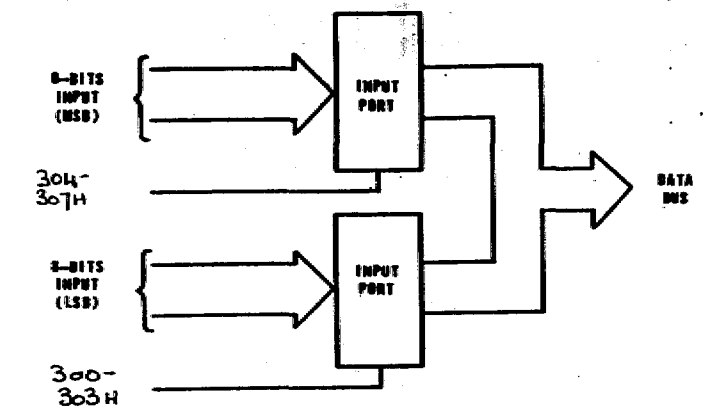


FIGURE 10-5. 16-bit Input Port

This program will print out the 16-bit value resulting from the combination of the two 8-bit values.

```
100  REM COMBINING TWO BYTES
110  REM
120  MSB = &H304
130  LSB = &H300
140  HIGH = INP (MSB)
150  LOW = INP (LSB)
160  VALUE = HIGH*256 + LOW
170  PRINT VALUE
```

Third Experiment: 8-bit digital-to-analogue converter

Parts required:

1 AD558 Analog Devices digital-to-analogue converter
1 16-pin wirewrap I.C. socket
1 0.1 microfarad capacitor

This experiment details the connection and programming of the AD558 (block diagram, fig. 10–6). This DAC has internal latches, built-in voltage reference and two-range voltage output. It is one of the easiest and cheapest DACs to interface without lowering quality. It needs a single +5 to +15V power supply.

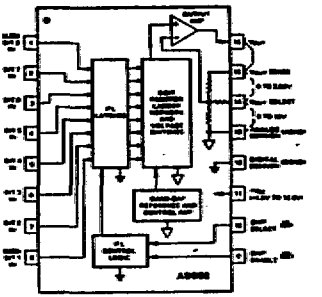


FIGURE 10-6. AD558 Functional Block Diagram

Connect the AD558 to the PC-35 according to fig. 10–7. This circuit will give a unipolar output of 0 to +10v. Binary data is sent to the DAC over the buffered data lines (BD7-BD0), and connected to the DAC's input lines. CHIP-ENABLE (CE-) is held low for transparent operation: i.e., any change in the input lines generates a corresponding instantaneous change in the output of the DAC. Under control of the OUT command, data is transferred to the device after being selected due to the CHIP SELECT (CS-) line going low. This is done by connecting the CS- line to one of the chip select lines on the PC-35. If the CS- line were connected to chip select line 0, the command to transfer data would be

OUT &H300, value.

If the CS- line were to be connected to chip select line 1, the command would be

OUT &H304, value.

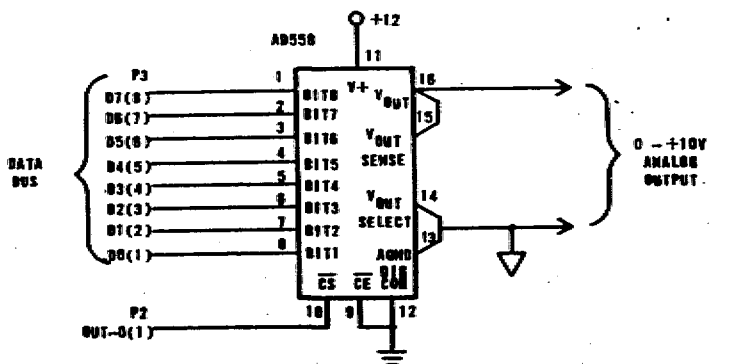


FIGURE 10-7. AD558 D/A Converter Interface

After wiring the DAC, it can be tested by writing data to it, and monitoring th analogue output with a voltmeter. Connect a DC voltmeter on the 0–10v range t the output of the DAC. As the DAC is an 8-bit device, only the values 0 to 255 ca be written to it. In these examples it is assumed that the DAC will be selected b address 300H. Different addresses will mean that the program must be change accordingly. The maximum value 255 can be written to the DAC with:

```
OUT &H300,255
```

The voltage at the DAC's output should be 9.96v. The output is not exactly 1 volts, because the highest possible output is

$$(255/256) * 10.0 = 9.96.$$

Most DAC specifications give instructions on how to adjust this error to get 10 maximum output.
To generate the smallest output, the command is

```
OUT &H300,0.
```

The voltmeter should read 0 volts. To get out of BASIC, enter the command

```
SYSTEM.
```

The output of the DAC will not change, because the internal latch on the AD55 maintains the value until the power is removed, or a new value is written to th DAC.
The following program will generate a voltage ramp, by incrementing, an decrementing, the value sent to the DAC.

```
100  REM  GENERATE VOLTAGE RAMP
110  REM
120  CLS
130  ADDR = &H300
140  FOR I = 0 TO 255
150       FOR J = 1 TO 10
160       NEXT J
170       OUT ADDR,I
180  NEXT I
190  FOR I = 255 TO 0 STEP -1
200       FOR J = 1 TO 10
210       NEXT J
220       OUT ADDR,I
230  NEXT I
240  GOTO 140
```

The next program generates random voltages on the DAC, and could be used to simulate random noise.

```
100  REM GENERATE RANDOM VOLTAGES
110  REM
120  ADDR = &H300
130  A = 256*RND(1)
140  PRINT A
150  OUT ADDR,A
160  REM
170  REM WASTING A BIT OF TIME
180  REM
190  FOR J = 1 TO 1000
200  NEXT J
210  GOTO 130
```

This program requires the entry of a voltage between 0 and +10 volts, and then calculates the number of bits to send to the DAC to generate the voltage closest to that entered.

```
100  REM   VOLTAGE TO BITS CONVERTER
110  REM
120  ADDR = &H300
130  INPUT "OUTPUT VOLTAGE (O TO 10) ?", VOLTS
140  BITS = INT((VOLTS/10) * 255)
150  PRINT "NUMBER OF BITS =", BITS
160  OUT ADDR,BITS
170  GOTO 130
```

Analogue signals can be generated with greater accuracy if a DAC with more than 8 bits of input is used, the data being sent to the DAC two bytes at a time, and some of these are shown.
The Analog Devices AD7522 is a 10-bit DAC with built-in double buffering (fig. 10–8). To write a value to it, a high buffer and a low buffer have to be loaded, followed by a buffer that dumps the high and low buffers to the DAC simultaneously. This makes for lengthy software, but protects the output from voltage irregularities.
The 10-bit A/D converter AD571 (fig. 10.9) is a similar device to the AD570. Its conversion can be started by using the same self-pulsing circuit, but two latches must be used to read the converted data.
Both of these A/D converters use the successive approximation principle. The other main type is the integrating converter, exemplified by the Intersil ICL7109. Fig. 10–10 shows its functional block diagram.

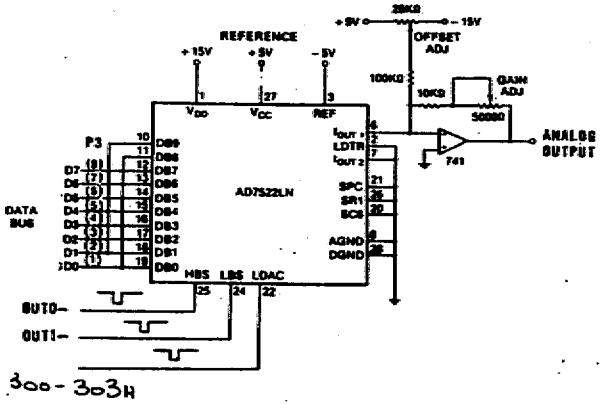


FIGURE 10-8. Schematic Diagram for a D/A Converter Interface Using the AD7522

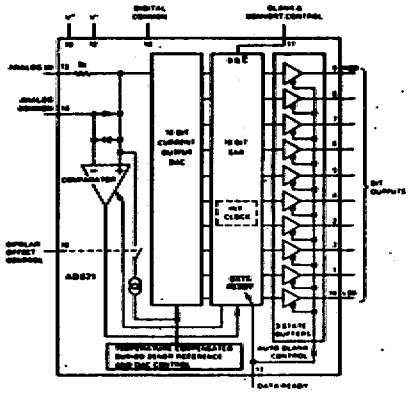


FIGURE 10-9 AD571 Functional Block Diagram

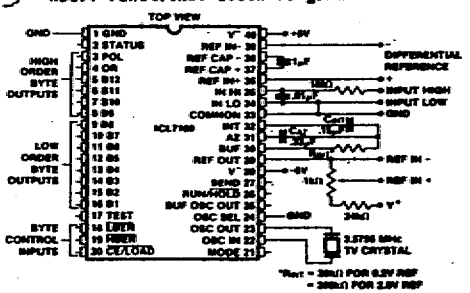


FIGURE 10-10. ICL 7109 Functional Block Diagram

ADDRESS RANGE		FUNCTION
DECIMAL	HEX	
0 - 63K	00000 - 0FFFF	16 - 64KB RAM, SYSTEM BOARD
64K - 255K	10000 - 3FFFF	192KB RAM, I/O CHANNEL
256K - 639K	40000 - 9FFFF	384KB RAM, I/O CHANNEL, FUTURE
640K - 655K	A0000 - A3FFF	RESERVED
656K - 703K	A4000 - AFFFF	
704K - 719K	B0000 - B3FFF	MONOCHROME
720K - 735K	B4000 - B7FFF	112KB GRAPHICS/DISPLAY VIDEO BUFFER
736K - 751K	B8000 - BBFFF	COLOR/GRAPHICS
752K - 767K	BC000 - BFFFF	
768K - 959K	C0000 - EFFFF	192KB ROM, MEMORY EXPANSION
960K - 975K	F0000 - F3FFF	RESERVED
976K - 1.024M	F4000 - FFFFF	48KB SYSTEM ROM

IBM PC
memory map

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 Registers
0Ax*	NMI Mask Register
0Cx	Reserved
0Ex	Reserved
100-1FF	Not Useable
200-20F	Game Control
210-217	Expansion Unit
220-24F	Reserved
278-27F	Reserved
2F0-2F7	Reserved
2F8-2FF	Asynchronous Communications (Secondary)
300-31F	Prototype Card
320-32F	Fixed Disk
378-37F	Parallel Printer
380-38F	SDLC Communications
3A0-3AF	Reserved
3B0-3BF	IBM Monochrome Display/Printer
3C0-3CF	Reserved
3D0-3DF	Color/Graphics
3E0-3E7	Reserved
3F0-3F7	Diskette
3F8-3FF	Asynchronous Communications (Primary)

*At power-on time, the Non Mask Interrupt (NMI) into the 8088 is masked off. This mask bit can be set and reset via system software as follows:

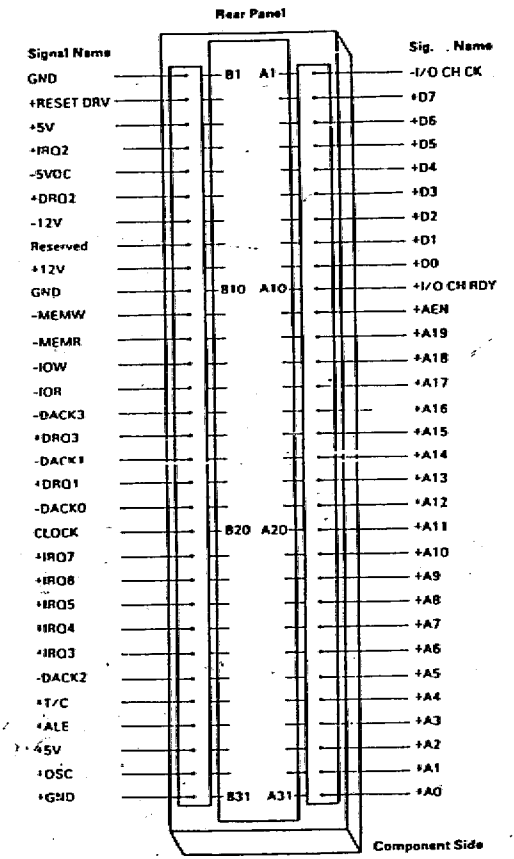
Set mask: write hex 80 to I/O Address A0 (enable NMI).

Clear mask: write hex 00 to I/O Address hex A0 (disable NMI).

I/O Address Map

B1

THE IBM-PC BUS CONNECTOR



I/O Channel Diagram

B2

I/O Channel Description

The following is a description of the IBM Personal Computer I/O Channel. All lines are TTL-compatible.

Signal	I/O	Description
OSC	O	Oscillator: High-speed clock with a 70-ns period (14.31818 MHz). It has a 50% duty cycle.
CLK	O	System clock: It is a divide by-three of the oscillator and has a period of 210 ns (4.77 MHz). The clock has a 33% duty cycle.
RESET DRV	O	This line is used to reset or initialize system logic upon power-up or during a low line voltage outage. This signal is synchronized to the falling edge of clock and is active high.
A0-A19	O	Address bits 0 to 19: These lines are used to address memory and I/O devices within the system. The 20 address lines allow access of up to 1 megabyte of memory. A0 is the least significant bit (LSB) and A19 is the most significant bit (MSB). These lines are generated by either the processor or DMA controller. They are active high.
D0-D7	I/O	Data Bits 0 to 7: These lines provide data bus bits 0 to 7 for the processor, memory, and I/O devices. D0 is the least significant bit (LSB) and D7 is the most significant bit (MSB). These lines are active high.
ALE	O	Address Latch Enable: This line is provided by the 8288 Bus Controller and is used on the system board to latch valid addresses from the processor. It is available to the I/O channel as an indicator of a valid processor address (when used with AEN). Processor addresses are latched with the failing edge of ALE.
$\overline{\text{I/O CH CK}}$	I	-I/O Channel Check: This line provides the processor with parity (error) information on memory or devices in the I/O channel. When this signal is active low, a parity error is indicated.

Signal	I/O	Description
I-O CH RDY	I	I-O Channel Ready: This line, normally high (ready), is pulled low (not ready) by a memory or I/O device to lengthen I/O or memory cycles. It allows slower devices to attach to the I/O channel with a minimum of difficulty. Any slow device using this line should drive it low immediately upon detecting a valid address and a read or write command. This line should never be held low longer than 10 clock cycles. Machine cycles (I/O or memory) are extended by an integral number of CLK cycles (210 ns).
IRQ2-IRQ7	I	Interrupt Request 2 to 7: These lines are used to signal the processor that an I/O device requires attention. They are prioritized with IRQ2 as the highest priority and IRQ7 as the lowest. An Interrupt Request is generated by raising an IRQ line (low to high) and holding it high until it is acknowledged by the processor (interrupt service routine).
$\overline{\text{IOR}}$	O	-I/O Read Command: This command line instructs an I/O device to drive its data onto the data bus. It may be driven by the processor or the DMA controller. This signal is active low.
$\overline{\text{IOW}}$	O	-I/O Write Command: This command line instructs an I/O device to read the data on the data bus. It may be driven by the processor or the DMA controller. This signal is active low.
$\overline{\text{MEMR}}$	O	Memory Read Command: This command line instructs the memory to drive its data onto the data bus. It may be driven by the processor or the DMA controller. This signal is active low.
$\overline{\text{MEMW}}$	O	Memory Write Command: This command line instructs the memory to store the data present on the data bus. It may be driven by the processor or the DMA controller. This signal is active low.

Signal	I/O	Description
DRQ1-DRQ3	I	DMA Request 1 to 3: These lines are asynchronous channel requests used by peripheral devices to gain DMA service. They are prioritized with DRQ3 being the lowest and DRQ1 being the highest. A request is generated by bringing a DRQ line to an active level (high). A DRQ line must be held high until the corresponding DACK line goes active.
$\overline{\text{DACK0}}$-$\overline{\text{DACK3}}$	O	-DMA Acknowledge 0 to 3: These lines are used to acknowledge DMA requests (DRQ1-DRQ3) and to refresh system dynamic memory (DACK0). They are active low.
AEN	O	Address Enable: This line is used to de-gate the processor and other devices from the I/O channel to allow DMA transfers to take place. When this line is active (high), the DMA controller has control of the address bus, data bus, read command lines (memory and I/O), and the write command lines (memory and I/O).
T/C	O	Terminal Count: This line provides a pulse when the terminal count for any DMA channel is reached. This signal is active high.

The following voltages are available on the system-board I/O channel:

+5 Vdc ± 5%, located on 2 connector pins
-5 Vdc ± 10%, located on 1 connector pin
+12 Vdc ± 5%, located on 1 connector pin
-12 Vdc ± 10%, located on 1 connector pin
GND (Ground), located on 3 connector pins

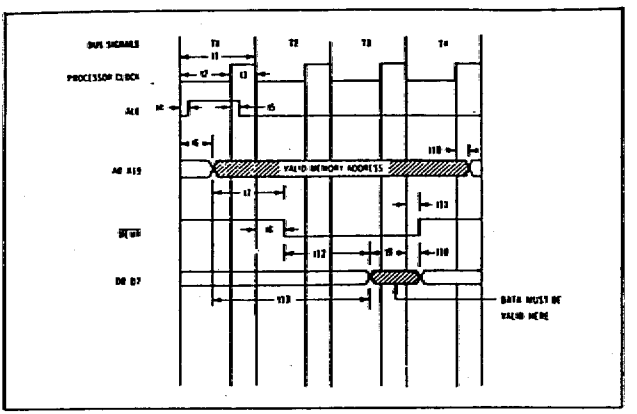


Fig. 6-1. Memory-read bus cycle timings.

Table 6-1. Memory-Read Bus Cycle Timings

Symbol	Max	Min
t1	–	209.5
t2	–	124.5
t3	–	71.8
t4	15	–
t5	15	–
t6	128	16
t7	–	91.5
t8	35	10
t9	–	42
t10	–	10
t11	35	10
t12	–	342
t13	–	458.5

*All times are in nanoseconds.

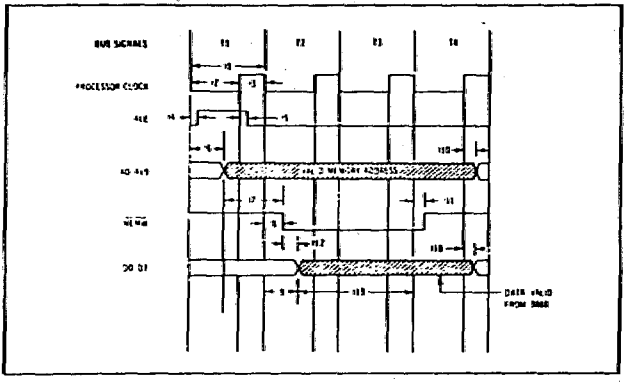


Fig. 6-2. Memory-write bus cycle timings.

Table 6-2. Memory-Write Bus Cycle Timings

Symbol	Max	Min
t1	–	209.5
t2	–	124.5
t3	–	71.8
t4	15	–
t5	15	–
t6	128	16
t7	–	91.5
t8	35	10
t9	122	14
t10	–	10
t11	35	10
t12	112	–
t13	–	297

*All timings are in nanoseconds.

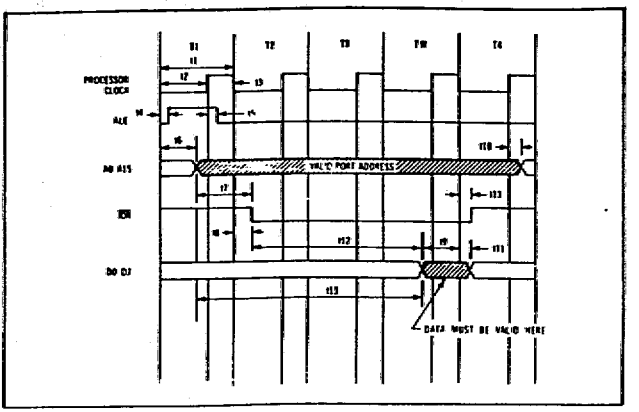


Fig. 6-3. I/O port read bus cycle timings.

Table 6-3. I/O Port Read Bus Cycle Timings

Symbol	Max	Min
t1	—	209.5
t2	—	124.5
t3	—	71.8
t4	15	—
t5	15	—
t6	128	16
t7	—	91.5
t8	35	10
t9	—	42
t10	—	10
t11	35	10
t12	—	551.5
t13	—	668

*All times are in nanoseconds.

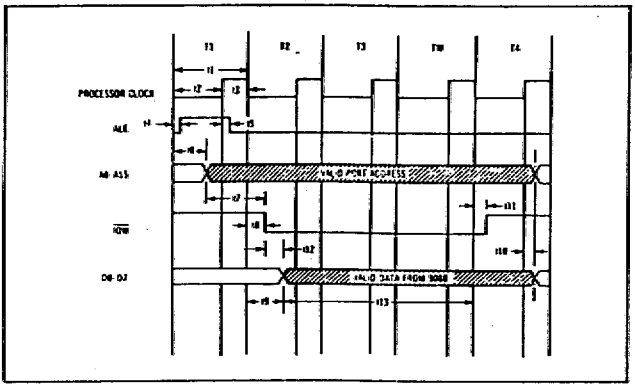


Fig. 6-4. I/O port write bus cycle timings.

Table 6-4. I/O Port Write Bus Cycle Timings

Symbol	Max	Min
t1	—	209.5
t2	—	124.5
t3	—	71.8
t4	15	—
t5	15	—
t6	128	16
t7	—	91.5
t8	35	10
t9	122	14
t10	—	10
t11	35	10
t12	112	—
t13	—	506.5

*All times are in nanoseconds.

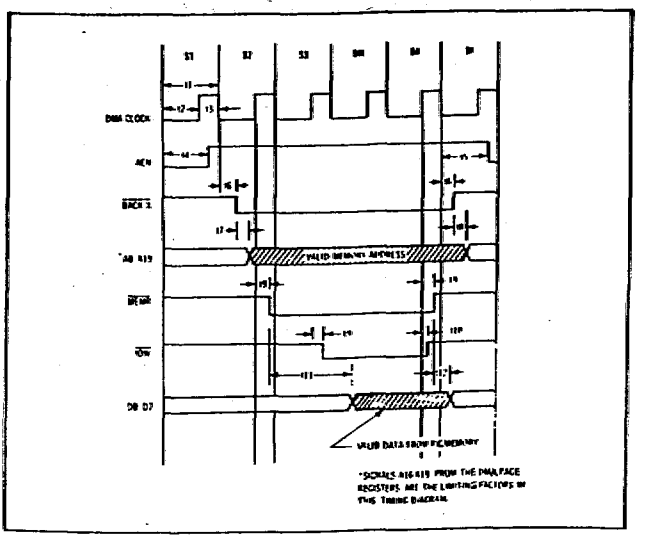


Fig. 6-5. DMA-initiated read-from-memory bus cycle timings.

Table 6-5. DMA Bus Cycle Timings

Symbol	Max	Min
t1	—	209.5
t2	—	119
t3	—	79
t4	183	132
t5	183	130
t6	170	—
t7	45	—
t8	—	11
t9	202	—
t10	142	—
t11	333	—
t12	—	4

*All times are in nanoseconds

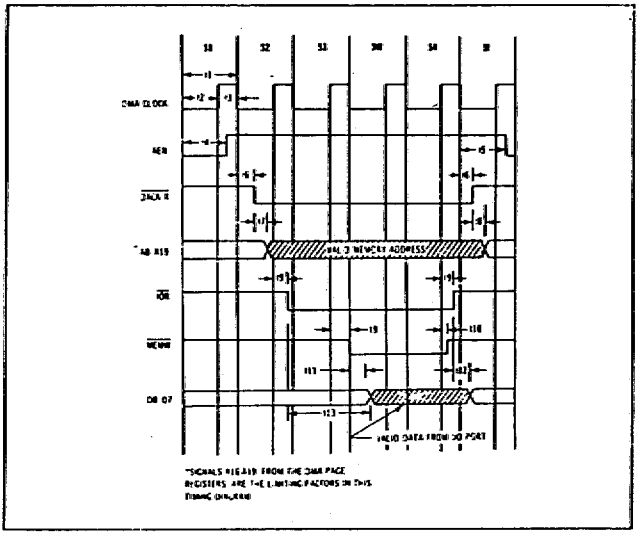
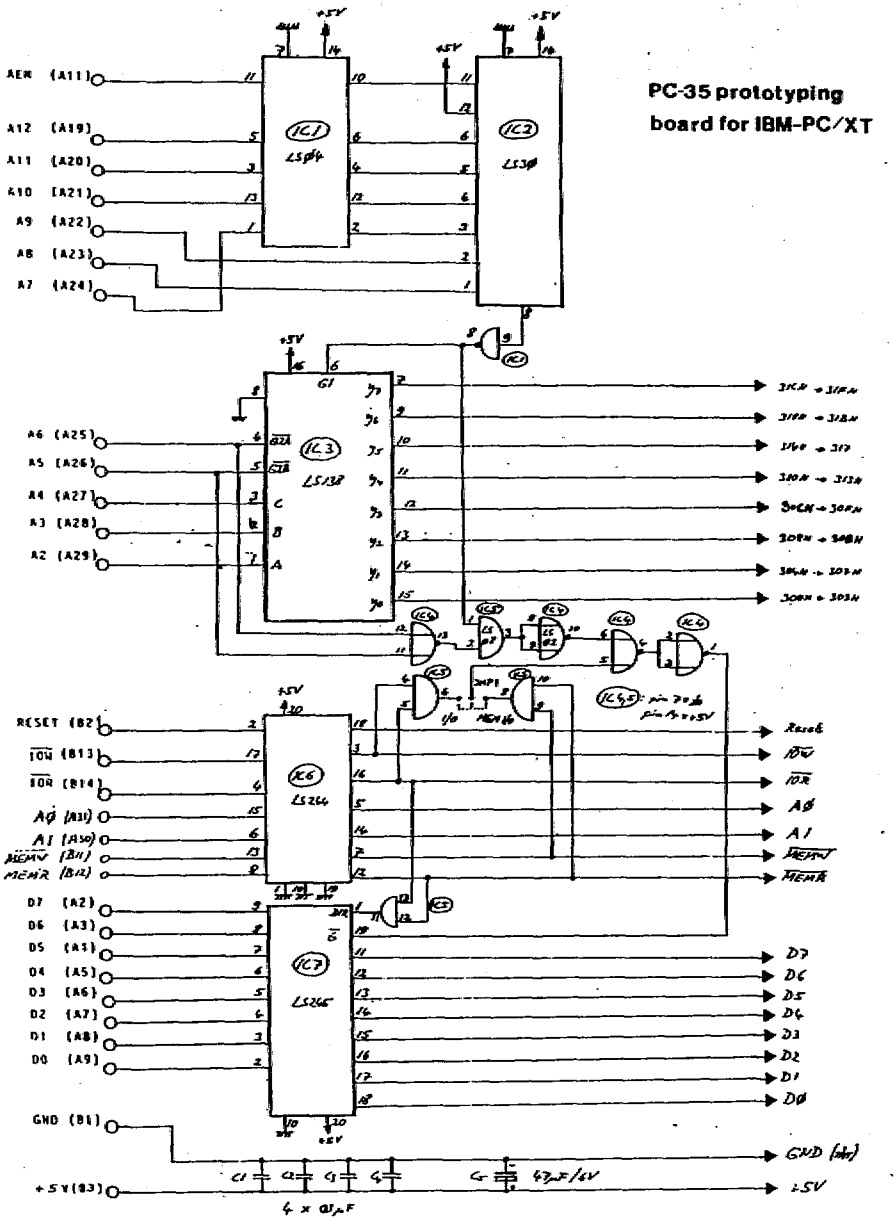


Fig. 6-6. DMA-initiated read-from-I/O bus cycle timings.

Table 6-6. More Bus Cycle Timings

Symbol	Max	Min
t1	—	209.5
t2	—	119
t3	—	79
t4	183	132
t5	183	130
t6	170	—
t7	45	—
t8	—	11
t9	202	—
t10	142	—
t11	30	—
t12	—	4
t13	240	—

*All times are in nanoseconds.


PC-35 prototyping
board for IBM-PC/XT
AEN (A11)
A12 (A19)
A11 (A20)
A10 (A21)
A9 (A22)
A8 (A23)
A7 (A24)
IC1
LS364
IC2
LS368
IC3
LS138
A6 (A25)
A5 (A26)
A4 (A27)
A3 (A28)
A2 (A29)
RESET (B2)
IOW (B13)
IOR (B14)
A0 (B31)
A1 (A30)
MEMW (B11)
MEMR (B12)
IC6
LS244
D7 (A2)
D6 (A3)
D5 (A4)
D4 (A5)
D3 (A6)
D2 (A7)
D1 (A8)
D0 (A9)
IC7
LS245
GND (B1)
+5V (B3)
4 x 0,1µF
47µF/6V
GND (chs)
+5V

Bibliography

(1) IBM Personal Computer Technical Reference Manual, IBM, 1981, Boca Raton, FL

(2) Tenley Design,"Inside the IBM PC," Starware, 1982, Washington, D.C.

Dooley G., Szybist D., "Accessing the Analog World," Chemical Engineering, August 22, 1983

Norton, H.K., Handbook of Transducers for Electronic Measuring Systems, Prentice-Hall, Englewood Cliffs, New Jersey, 1969

Hansen J., "Micros are Sounding Good," Microcomputing, August 1982

Sargent M., Shoemaker R.L., "Interfacing Microcomputer to the Real World," Addison-Wesley, Reading, Massachusetts,1981

Ciarcia S.,"Add Programmable Sound Effects to Your Computer," Byte, July 1982

Ciarcia S., "Memory Mapped IO," Byte, November 1977

Grant G., "Interfacing the AD558 DACPORT to Microprocessors," Analog Devices Application Note, 1982

Jaeger R.C.,"Tutorial: Analog Data Acquisition Technology," Part 1, IEEE Micro, 1982

Ciarcia S., "Build a Low-Cost Speech Synthesizer Interface," Byte, June 1981

Ciarcia S., "Build an Unlimited-Vocabulary Speech Synthesizer," Byte, September 1982

Ciarcia S., "Analog Interfacing in the Real World," Byte, January 1982

Herceg E.E., "Handbook of Measurement and Control," Schaevitz Engineering, Pennsauken, N.J., 1976

Barden Jr W., "Putting Real-World Interfaces to Work, Part 1: Monitoring Physical Quantities with the TRS-80," Byte, October 1982

Hallgren R.C., "Interface Projects for the TRS-80," Prentice-Hall, Englewood Cliffs, N.J.,1982

Ciarcia S.,"Everyone Can Know the Real Time," Byte, May 1982

Tocci R.J., Laskowski L.P., "Microprocessors and Microcomputers: Hardware and Software," Prentice-Hall, Englewood Cliffs, N.J., 1979

Larsen D.G., Rony P.R., Titus J.A., "Microcomputer I/O Devices," American Laboratory, November 1975

Sheingold D.H., "Analog - Digital Conversion Notes," Analog Devices, Inc., Norwood, Massachusetts 1977

Ciarcia S.," Ciarcia's Circuit Cellar ,"Byte Books, Peterborough, N.H., 1979

Zaks R., Lesea A., "Microprocessor Interfacing Techniques," Sybex, Berkeley, C.A., 1979

Cameron J., "A High-Resolution Analog-to-Digital Converter for the TRS-80," Byte, February 1982.

Cosgrove D., "The Microcomputer as a Laboratory Instrument," Byte, November 1981

Balcom D.,"Interfacing a Microcomputer to the Analog World." Interface Age, July 1978

Olsen H.,"Controlling the Real World." Byte., March 1978

OK Industries, Inc., Catalog 82-36P, 1982, Bronx, NY

Titus J., Rony P., Titus C., Larsen D., "Microcomputer-Analog Converter Software and Hardware Interfacing." Howard W. Sams., Indianapolis,. I.N. 1978