I/O Interface

In this chapter, the basic and programmable Input/Output interfaces are introduced. This provides details on parallel and serial interfacing.

Objectives

- Upon completion of this chapter, you will be able to
  - Explain the operation of the basic input and output interfaces.
  - Decode an 8-, 16- and 32-bit I/O device so that it can be used at any I/O port address.
  - Define handshaking and explain how to use it with I/O devices.
  - Interface and program the 82C55 programmable parallel interface.
  - Interface and program the 16550 serial communications interface adapter.

Introduction to I/O Interface

- Both the IN and OUT instructions transfer data between an I/O device and the microprocessor's accumulator (AL, AX, or EAX).
- The I/O address is stored in
  - register DX as a 16-bit I/O address or
  - the byte immediately following the opcode as an 8-bit I/O address.

Introduction to I/O Interface

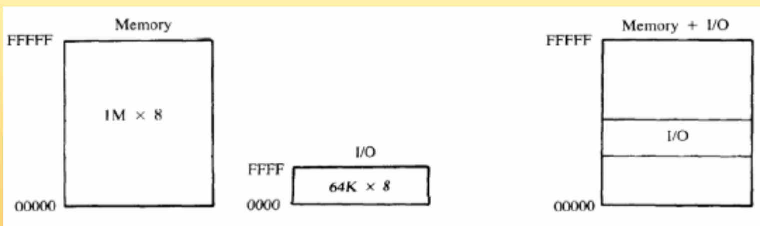
- The I/O address is usually called a port number.
- The INS and OUTS instructions are also provided in all Intel microprocessors except 8086/8088 to transfer strings of data between the memory and an I/O device.

I/O Instructions

Instruction	Data Width	Function
IN AL,p8	8	A byte is input from port p8 into AL
IN AX,p8	16	A word is input from port p8 into AX
IN EAX, p8	32	A doubleword is input from port p8 into EAX
IN AL,DX	8	A byte is input from the port addressed by DX into AL
IN AX,DX	16	A word is input from the port addressed by DX into AX
IN EAX,DX	32	A word is input from the port addressed by DX into EAX
INSB	8	A byte is input from the port addressed by DX into the extra segment memory location addressed by DI, then DI = DI ± 1
INSW	16	A word is input from the port addressed by DX into the extra segment memory location addressed by DI, then DI = DI ± 2
INSD	32	A doubleword is input from the port addressed by DX into the extra segment memory location addressed by DI, then DI ± 4
OUT p8,AL	8	A byte is output from AL to port p8
OUT p8,AX	16	A word is output from AX to port p8
OUT p8,EAX	32	A doubleword is output from EAX to port p8
OUT DX,AL	8	A byte is output from AL to the port addressed by DX
OUT DX,AX	16	A word is output from AX to the port addressed by DX
OUT DX,EAX	32	A doubleword is output from EAX to the port addressed by DX
OUTSB	8	A byte is output from the data segment memory location addressed by SI to the port addressed by DX, then SI = SI ± 1
OUTSW	16	A word is output from the data segment memory locations addressed by SI to the port addressed by DX, then SI = SI ± 2
OUTSD	32	A doubleword is output from the data segment memory locations addressed by SI to the port addressed by DX, then SI = SI ± 4

Isolated and Memory-mapped I/O

- There are two completely different methods of interfacing I/O to the microprocessor: isolated I/O and memory-mapped I/O.



(a) Isolated I/O (b) Memory-mapped I/O

The memory and I/O maps for the 8086/8088 microprocessors.

Isolated and Memory-mapped I/O

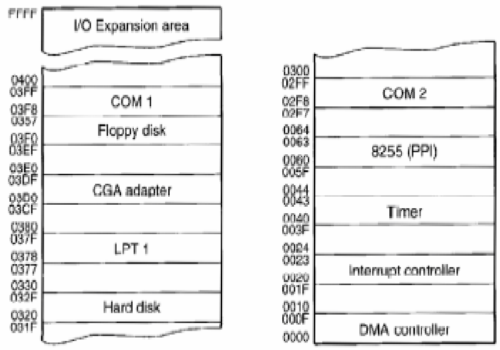
- Isolated I/O:
  - I/O locations are isolated from the memory system in a separate I/O address space.
  - The addresses for isolated I/O devices are usually called ports.
- Separate control signals for the I/O space are developed using $M/\overline{IO}$ and $W/\overline{R}$ to indicate an I/O read ($\overline{IORC}$) or an I/O write ($\overline{IOWC}$) operation.
- Advantage: The memory can be expanded to its full size without using any of its space for I/O devices.
- Disadvantage: The amount of circuitry required increases.

Isolated and Memory-mapped I/O

- Memory-mapped I/O:
  - A memory-mapped I/O device is treated as a memory location in the memory map.
  - No separate I/O interface, separate I/O instructions, separate I/O control signals such as $\overline{IORC}$ and $\overline{IOWC}$ is required.
- Advantage: Any memory transfer instruction can be used to access the I/O device.
- Disadvantage: A portion of the memory system is used as the I/O map.

The PC I/O Map

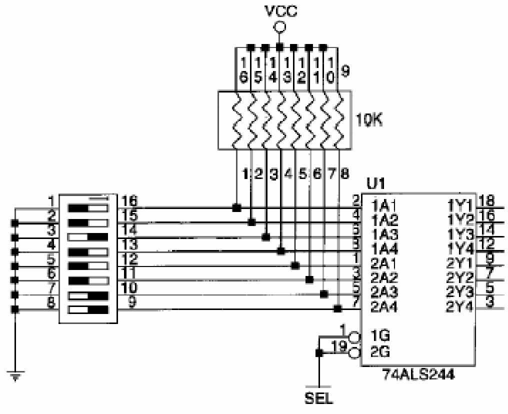
- The personal computer uses part of the I/O map for dedicated functions.



The I/O map of a personal computer illustrating many of the fixed I/O areas

Basic I/O Interfaces

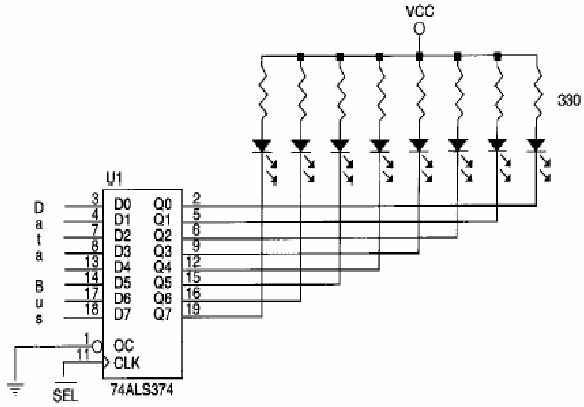
- The basic input device is a set of 3-state buffers.



The basic input interface illustrating the connection of eight switches. Note that the 74ALS244 is a three-state that controls the application of the switch data to the data bus.

Basic I/O Interfaces

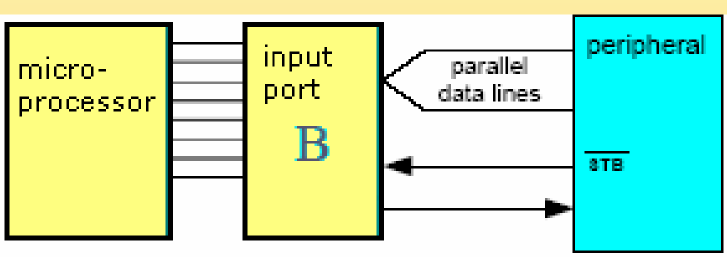
- The basic output device is a set of data latches.



The basic output interface connected to a set of LED displays

Handshaking

- Many I/O devices accept or release information at a much slower rate than the microprocessor.
- Handshaking is a mechanism to synchronize the I/O device with the microprocessor.



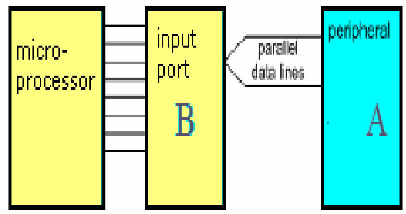
Signal directions for handshake input data transfer

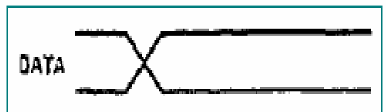
Handshaking

Case (a)

1. A puts the datum on the bus.

No one cares whether B is ready or even exist or not.





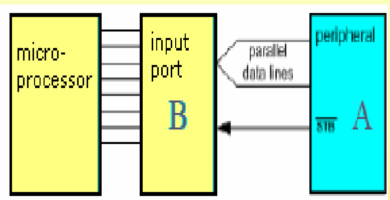
Simple Input

Handshaking

Case (b)

1. A puts the datum on the bus.
2. A says, "Here is a datum for you."

After few ms, A drops the datum no matter whether B gets it or not.



Signal directions for handshake input data transfer

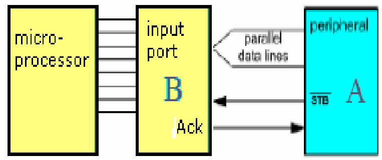
(b) Simple strobe I/O.

Handshaking

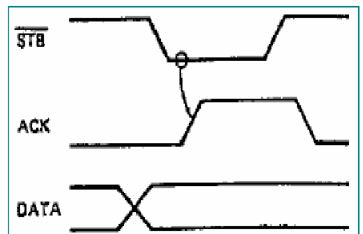
Case (c)

1. A puts the datum on the bus.
2. A says, “Here is a datum for you.”
3. B says, “I got it. Send me another.”

Note A holds the datum until it is acknowledged by B.



Signal directions for handshake input data transfer



(c) Single handshake I/O.

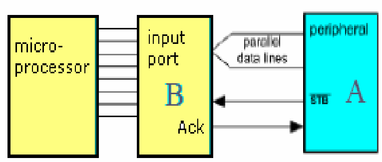
Handshaking

Case (d)

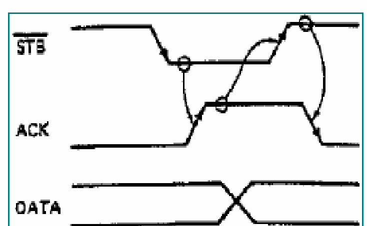
1. A asks, “Are you ready?”
2. B says, “I’m ready.”
3. A puts the datum on the bus.
4. A says, “Here is the datum for you.”
5. B says, “I got it. Thank you.”

Note A holds the datum until it is acknowledged by B.

Since A doesn’t send the datum unless he makes sure that B is ready, B won’t keep A holding a datum for a long time.



Signal directions for handshake input data transfer



(d) Double handshake I/O.

Address Decoding

- I/O address decoding is
  - exactly the same as memory address decoding for memory-mapped I/O devices, and is
  - very similar to memory address decoding for isolated I/O devices.
- The main difference between memory decoding and isolated I/O decoding is
  - the number of address pins connected to the decoder and
  - that we use the $\overline{IORC}$ and $\overline{IOWC}$ generated with $M/\overline{IO}=0$ or $IO/\overline{M}=1$ to activate I/O devices for a read or a write operation.

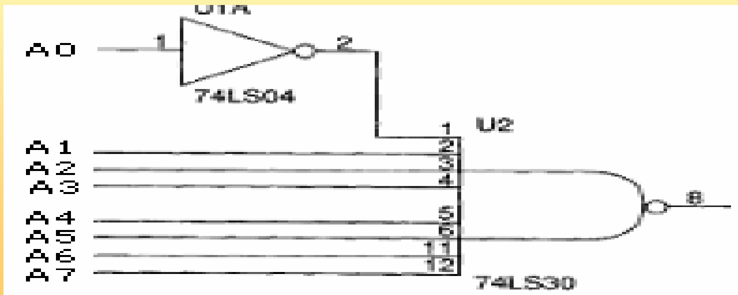
Address Decoding Techniques

- In order to attach an I/O port or a memory device to the microprocessor, it is necessary to decode the address sent from the microprocessor.
- A few more common address-decoding techniques will be described here, namely:
- Simple logic gate decoder
- Non-programmable decoder
- EPROM decoder
- PLD Programmable Decoder

Simple logic gate decoder

8-bit I/O address

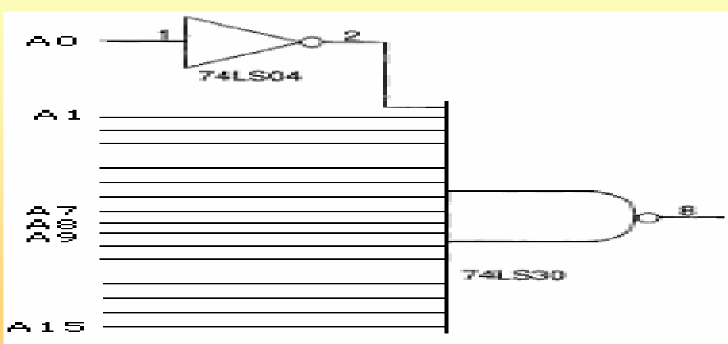
If the system will never contain more than 256 I/O devices, we often decode only address connections A7-A0 for an 8-bit I/O port address.



Address: 1111 1110B = FEH

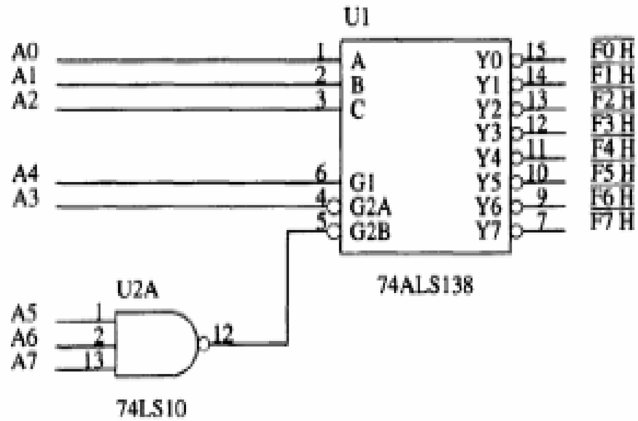
Simple logic gate decoder

16-bit I/O address



Address = 1111 1111 1111 1110B = FFFEH

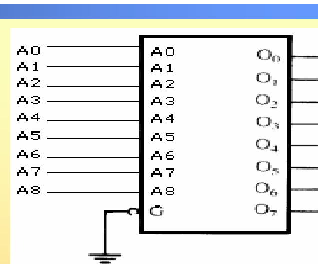
Non-programmable decoder



A port decoder that decodes 8-bit I/O ports. This decoder generates active low outputs for ports F0H–F7H.

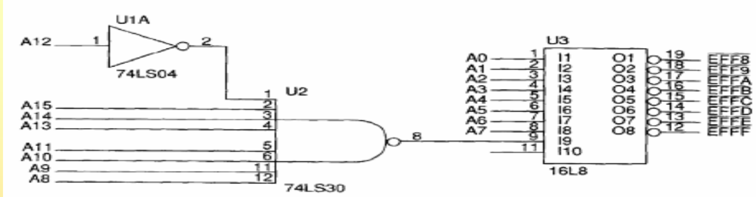
EPROM decoder

Unlike the 74LS138, the EPROM has many input connections, which reduces the number of other circuits required in a system memory address decoder.



Inputs										Outputs							
$\overline{G}$	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7
0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1
0	0	0	1	1	1	1	0	0	1	1	0	1	1	1	1	1	1
0	0	0	1	1	1	1	0	1	0	1	1	0	1	1	1	1	1
0	0	0	1	1	1	1	0	1	1	1	1	1	0	1	1	1	1
0	0	0	1	1	1	1	1	0	0	1	1	1	1	0	1	1	1
0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	0	1	1
0	0	0	1	1	1	1	1	1	0	1	1	1	1	1	1	0	1
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
All other combinations										1	1	1	1	1	1	1	1

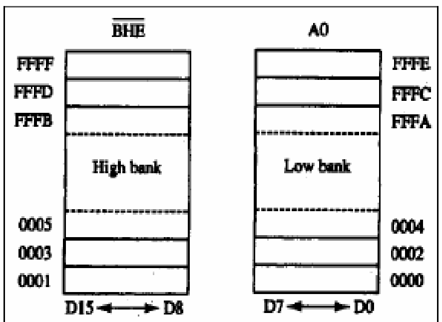
PLD programmable decoder




```
AUTHOR     Barry B. Brey
COMPANY    BreyCo
DATE       7/2/96
CHIP       DECODER9 PAL16L8
;pins  1  2  3  4  5  6  7  8  9    10
       A0 A1 A2 A3 A4 A5 A6 A7 NAND GND

;pins 11  12    13    14    15    16    17    18    19    20
      NC  EFFFH EFFEH EFFDH EFFCH EFFBH EFFAH EFF9H EFF8H VCC

EQUATIONS
/EFF8H = A7 * A6 * A5 * A4 * A3 * /A2 * /A1 * /A0 * /NAND
/EFF9H = A7 * A6 * A5 * A4 * A3 * /A2 * /A1 * A0 * /NAND
/EFFAH = A7 * A6 * A5 * A4 * A3 * /A2 * A1 * /A0 * /NAND
/EFFBH = A7 * A6 * A5 * A4 * A3 * /A2 * A1 * A0 * /NAND
/EFFCH = A7 * A6 * A5 * A4 * A3 * A2 * /A1 * /A0 * /NAND
/EFFDH = A7 * A6 * A5 * A4 * A3 * A2 * /A1 * A0 * /NAND
/EFFEH = A7 * A6 * A5 * A4 * A3 * A2 * A1 * /A0 * /NAND
/EFFF7 = A7 * A6 * A5 * A4 * A3 * A2 * A1 * A0 * /NAND
```


8- and 16-bit I/O ports

The 8086, 80186, 80286 or 80386SX I/O system contains two 8-bit banks, just as memory does.



The I/O banks found in the 8086, 80186, 80286, and 80386SX

8- and 16-bit Data Transfer

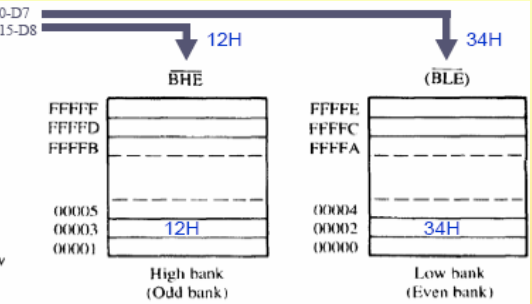
Example:
Address =
0000 0000 0000 0000 001-

Case 1: $\overline{BHE}$=0, $\overline{BLE}$=1

Case 2: $\overline{BHE}$=1, $\overline{BLE}$=0

Case 3: $\overline{BHE}$=0, $\overline{BLE}$=0

Note: A_0 is labeled $\overline{BLE}$ (Bus low enable) on the 80386SX.



The high (odd) and low (even) 8-bit memory banks of the 8086/8028J6/83865X microprocessors.

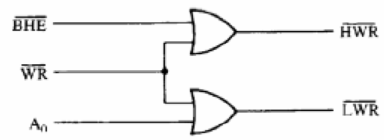
$\overline{BHE}$	$\overline{BLE}$ (A_0)	Function
0	0	Both banks enabled for a 16-bit transfer
0	1	High bank enabled for an 8-bit transfer
1	0	Low bank enabled for an 8-bit transfer
1	1	No banks enabled

8- and 16-bit I/O ports

- Any 8-bit I/O write requires a separate write strobe to function correctly.
- I/O reads do not require separate I/O read strobes as the microprocessor reads only the byte it expects and ignores the other byte.

Separate Bank Write Strobes

- It's more effective and less expensive.

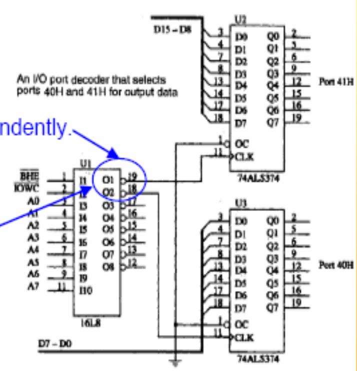



```
TITLE        Address Decoder
CHIP         DECODER2  PAL16L8

;pins 1   2   3   4   5   6   7   8   9  10
     A23 A22 A21 A20 A19 A18 A17 A16 A0 GND

;pins 11  12  13  14  15 16 17   18 19 20
      BHE SEL LWR HWR NC NC MWTC NC NC VCC

EQUATIONS

/SEL = /A23 * /A22 * /A21 * /A20 * /A19 * A18 * A17 * /A16
/LWR = /MWTC * /AO
/HWR = /MWTC * /BHE
```


8-bit wide ports

- Example : 2 different 8-bit output devices located at 8-bit I/O address 40H and 41H.

The 2 8-bit ports are controlled independently.



EXAMPLE

```
CHIP     DECODERA PAL16L8

;pins 1      2    3  4  5  6  7  8  9  10
     BHE IOWC A0 A1 A2 A3 A4 A5 A6 GND

;pins 11 12 13 14 15 16 17 18 19 20
     A7 NC NC NC NC NC NC 40 41 VCC

EQUATIONS

/40 = /AO  * /IOWC * /A7 * A6 * /A5 * /A4 * /A3 * /A2 * /A1
/41 = /BHE * /IOWC * /A7 * A6 * /A5 * /A4 * /A3 * /A2 * /A1
```

16-bit wide ports

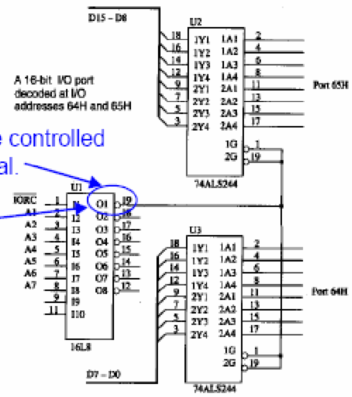
- Example :

A 16-bit input device connected to function at 8-bit I/O addresses 64H and 65H.

EXAMPLE

```
AUTHOR    Barry B. Brey
COMPANY   BreyCo
DATE      7/5/96

CHIP      DECODERB PAL16L8

;pins  1   2   3   4   5   6   7   8   9   10
     IORC A1  A2  A3  A4  A5  A6  A7  NC  GND

;pins 11  12  13  14  15  16  17  18  19   20
      NC  NC  NC  NC  NC  NC  NC  NC  O6X  VCC

EQUATIONS

/O6X = /IORC * /A7 * A6 * A5 * /A4 * /A3 * /A2 * /A1
```

The 2 8-bit ports are controlled with one single signal.

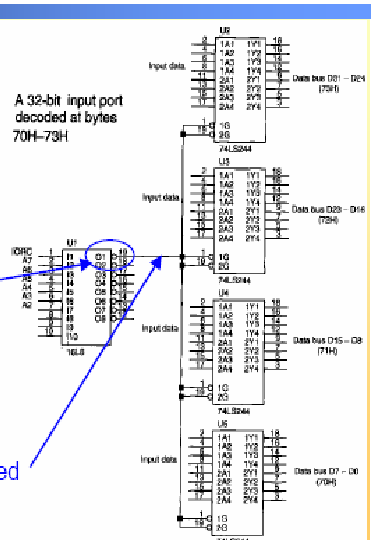


32-bit wide ports

EXAMPLE

```
AUTHOR    Barry B. Brey
COMPANY   BreyCo
DATE      7/6/96
CHIP      DECODERC PAL16L8

;pins   1   2   3   4   5   6   7   8   9   10
     IORC  A7  A6  A5  A4  A3  A2  NC  NC  GND

;pins 11  12  13  14  15  16  17  18  19   20
      NC  NC  NC  NC  NC  NC  NC  NC  SEL  VCC

EQUATIONS

/SEL = /IORC * /A7 * A6 * A5 * A4 * /A3 * /A2
```

A 32-bit input port decoded at bytes 70H–73H



The 4 8-bit ports are controlled with one single signal.