CMPE 310
Systems Design and Programming
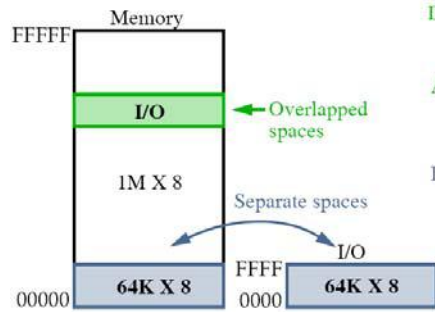
L : Chapter 11 – BASIC I/O Interface

UMBC
AN HONORS UNIVERSITY IN MARYLAND

L9 Objectives

* Utilize basic I/O instructions
* I/O port decoding
* Diagram how I/O devices are connected to the x86/88

Isolated versus Memory-mapped I/O

* In the Isolated scheme, IN, OUT, INS and OUTS are required.
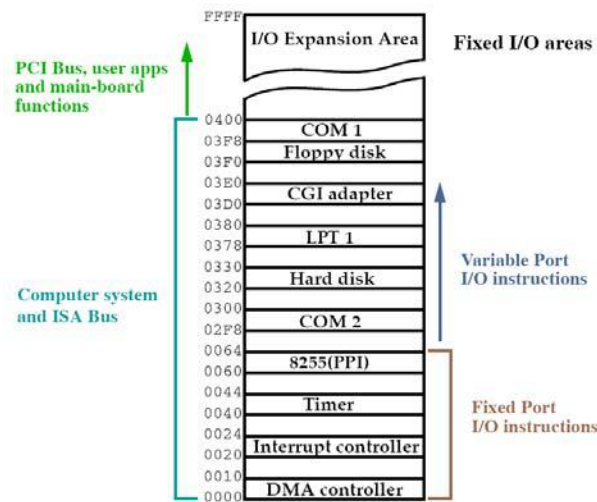* In the Memory-mapped scheme, any instruction that references memory can be used

* 8-bit port addresses used to access system board device, e.g. timer and keyboard interface.
* 16-bit port addresses used to access serial and parallel ports, hard drives, etc.

Personal computer I/O MAP

Basic I/O instructions

* *IN* and *OUT* 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 (variable addressing).
    * The byte (p8) immediately following the opcode (fixed address).

  * *IN   AL, 19H*   ;8-bits are saved to AL from I/O port 19H.
  * *IN   EAX, DX*   ;32-bits are saved to EAX.
  * *OUT DX, EAX*   ;32-bits are written to port DX from EAX.
  * *OUT 19H, AX*   ;16-bits are written to I/O port 19H.

  * Only 16-bits (A_0 to A_{15}) are decoded.
    * Address connections above A_{15} are undefined for I/O instructions.
    * Decoded at 0000H-03FFH on the ISA bus.
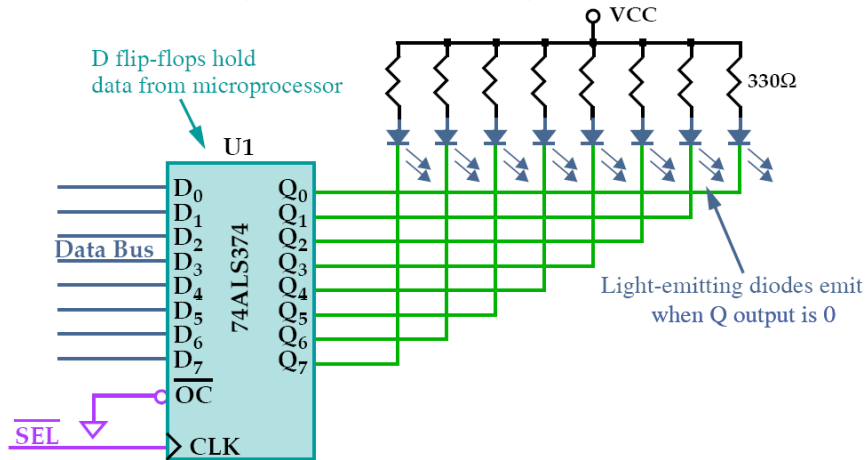 * INS and OUTS transfer data strings to I/O devices using ES:DI and DS:SI, respectively.

Basic input interface

* The basic input device (to the microprocessor) is a set of tri-state buffers.
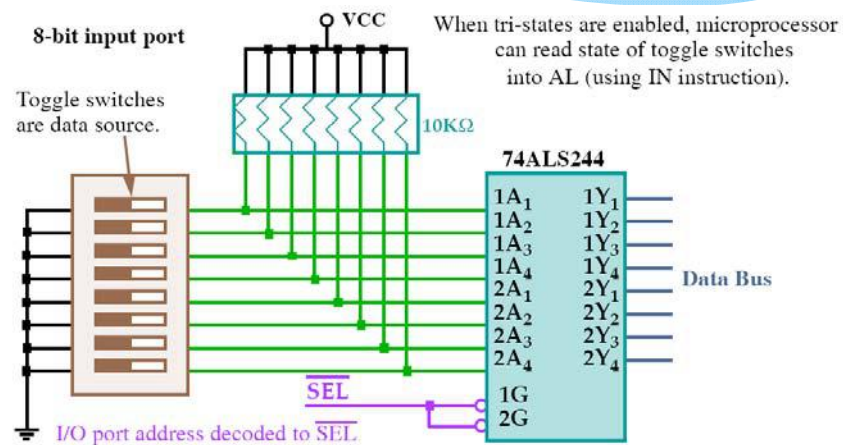
Basic output interface

The basic output device (from the microprocessor) is a set of latches.

In this case, the data from the OUT instruction is latched using SEL.

Handshaking

* I/O devices are typically slower than the microprocessor.
* Handshaking is used to synchronize I/O with the microprocessor.
  * A device indicates that it is ready for a command or data (through some I/O pin or port).
  * The processor issues a command to the device, and the device indicates it is busy (not ready).
  * The I/O device finishes its task and indicates a ready condition, and the cycle continues.
* There are two basic mechanisms for the processor to service a device.
  * *Polling: Processor initiated.* Device indicates it is ready by setting some status bit and the processor periodically checks it.
  * *Interrupts: Device initiated.* The act of setting a status bit causes an interrupt, and the processor calls an ISR to service the device.
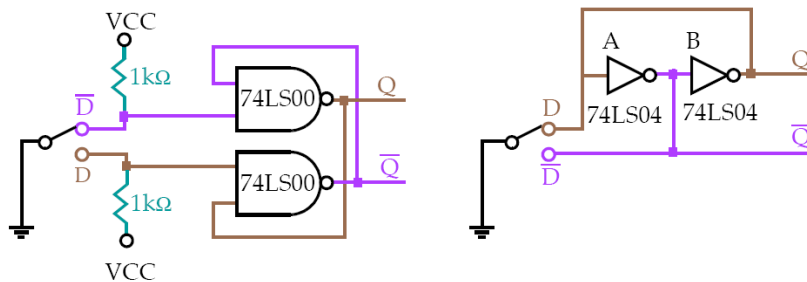
Interfacing circuitry

* The terminal characteristics of the processor must be matched to those of the I/O devices.
* *Input Devices:*
    * They are either:
        * TTL (0.0V-0.8V low and 2.0-5.0V high) or compatible.
        * Switch-based; usually either open or connected.
            * These must be conditioned before they can be used properly.
            * For example, to make a simple (single-pole, single-throw) toggle switch TTL compatible:

VCC

2.2kΩ

TTL Output

SPST
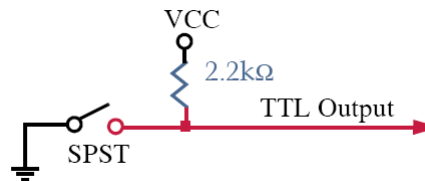
    * The value of R can vary between 1K and 10KΩ.

Interfacing circuitry

* *Input Devices:*
* Mechanical switches physically bounce when they are closed (causing them to momentarily open after being closed).
    * This can cause a problem if they are used as a clocking signal.
* Two asynchronous flip-flop solutions are given below:
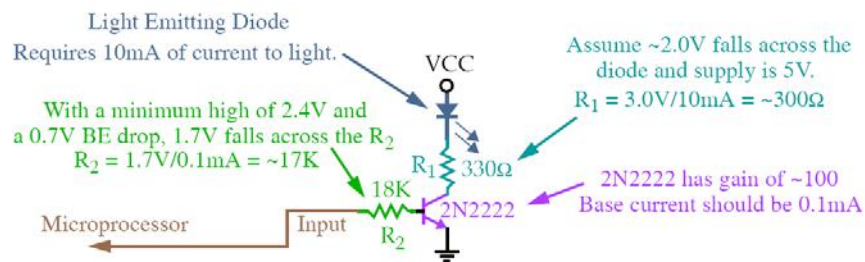
VCC

1kΩ

$\overline{D}$

D

74LS00

74LS00

Q

$\overline{Q}$

1kΩ

VCC

A

B

Q

D

74LS04

74LS04

$\overline{D}$

$\overline{Q}$

* The basic idea is that these flip-flops store the values even if the D/$\overline{D}$ nodes both float.
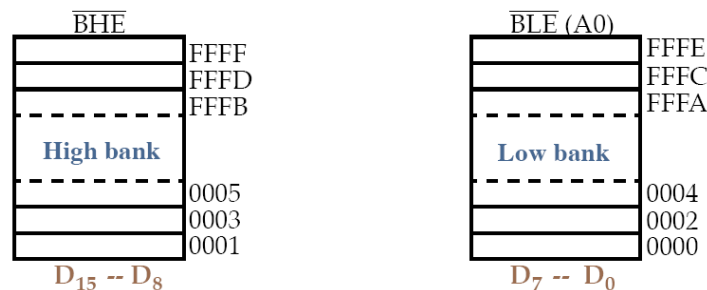
Interfacing circuitry

* **Output Devices:**
  * Interfacing an output device requires matching the voltage and current relationships of the devices and processor.
  * Remember that the standard output levels of TTL compatible devices are 0.0 to 0.4V for logic 0 and 2.4V to 5.0V for logic 1.
 * The current levels are 0.0 to 2.0mA (logic 0) and 0.0 to -400uA (logic 1).

  For example:

Light Emitting Diode
Requires 10mA of current to light.

VCC

Assume ~2.0V falls across the diode and supply is 5V.
$R_1 = 3.0V/10mA = \sim 300\Omega$

With a minimum high of 2.4V and a 0.7V BE drop, 1.7V falls across the R_2
$R_2 = 1.7V/0.1mA = \sim 17K$

$R_1 \gtrless 330\Omega$

18K

2N2222

2N2222 has gain of ~100
Base current should be 0.1mA

Microprocessor   Input R_2

I/O port decoding

* For memory-mapped I/O, decoding is identical to memory decoding.
* For isolated I/O, $\overline{\text{IORC}}$ and $\overline{\text{IOWC}}$ are developed using $M/\overline{\text{IO}}$ and $W/\overline{R}$ pins of the microprocessor.
  * The text gives examples of 8-bit decoding and 16-bit decoding, which is a straightforward application of devices we've used for memory decoding.
* The I/O banks on the 8086 through the 80386SX are also set up like the memory.

$\overline{\text{BHE}}$

FFFF
FFFD
FFFB

High bank

0005
0003
0001

$D_{15} -- D_8$

$\overline{\text{BLE}}$ (A0)

FFFE
FFFC
FFFA

Low bank

0004
0002
0000

$D_7 -- D_0$

I/O port decoding

* Similar to memory writes, any 8-bit I/O write request requires separate write strobes ($\overline{BHE}$ and $\overline{BLE}$) but read requests do not.
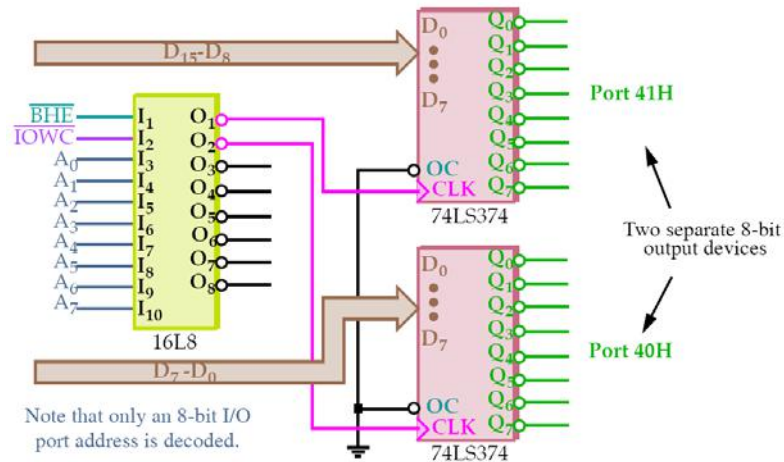
I/O port decoding

* Output devices can be 16-bit in which case $\overline{BHE}$ is not needed.

* Input devices can be 8-bit or 16-bit.
    * Note that instead of latches, high impedance buffers (74ALS244) are used in these cases.

* 32-bit ports are becoming more popular because of PCI bus primarily.
    * The EISA and VESA local bus are also 32-bit buses.

* For the 64-bit data buses of the Pentium, the I/O ports can appear in any of the 8 banks.
    * However, only 32-bit transfers are supported, as there are no 64-bit transfer instructions.

Example

* Write an assembly language instruction that send a byte (36H) of data to a fixed port address of 43H.

* Transfer content of BL to port 378H

Next Time

* IO Interface
  * 82C55 Programmable Peripheral Interface

STOP