**Basic I/O Instructions**

We discussed IN, OUT, INS and OUTS as instructions for the transfer of data to and from an I/O device.

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 0019H.
```

Only 16-bits ($A_0$ to $A_{15}$) are decoded.

Address connections above $A_{15}$ are undefined for I/O instructions.

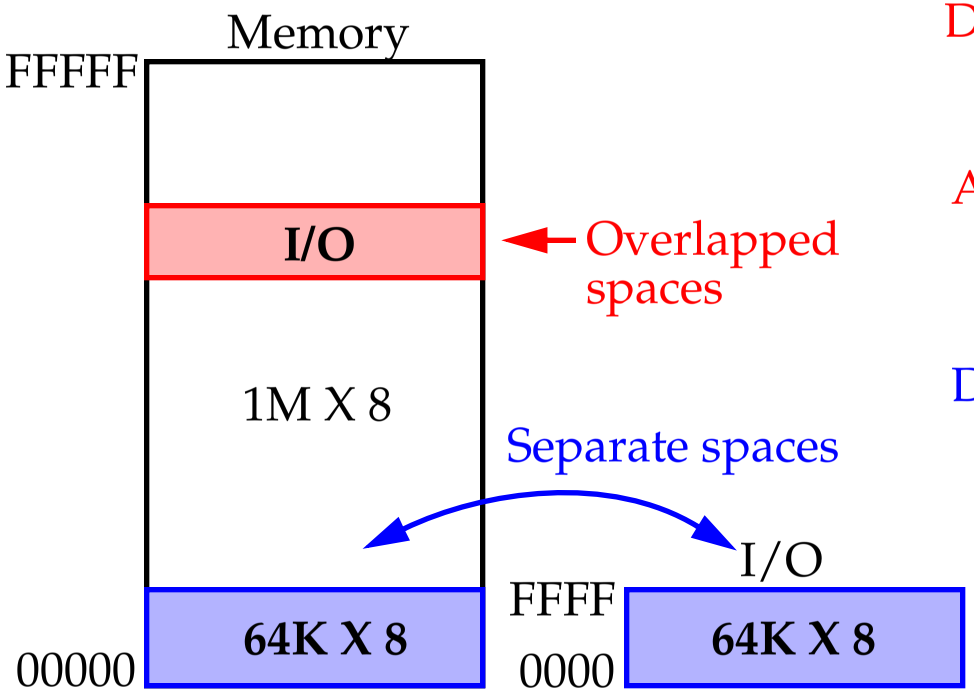
0000H-03XXH are used for the ISA bus.

INS and OUTS transfer to I/O devices using ES:DI and DS:SI, respectively.

**Isolated versus Memory-Mapped I/O**

Isolated and 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.




Disadvantage:
A portion of the memory space is used for I/O devices.

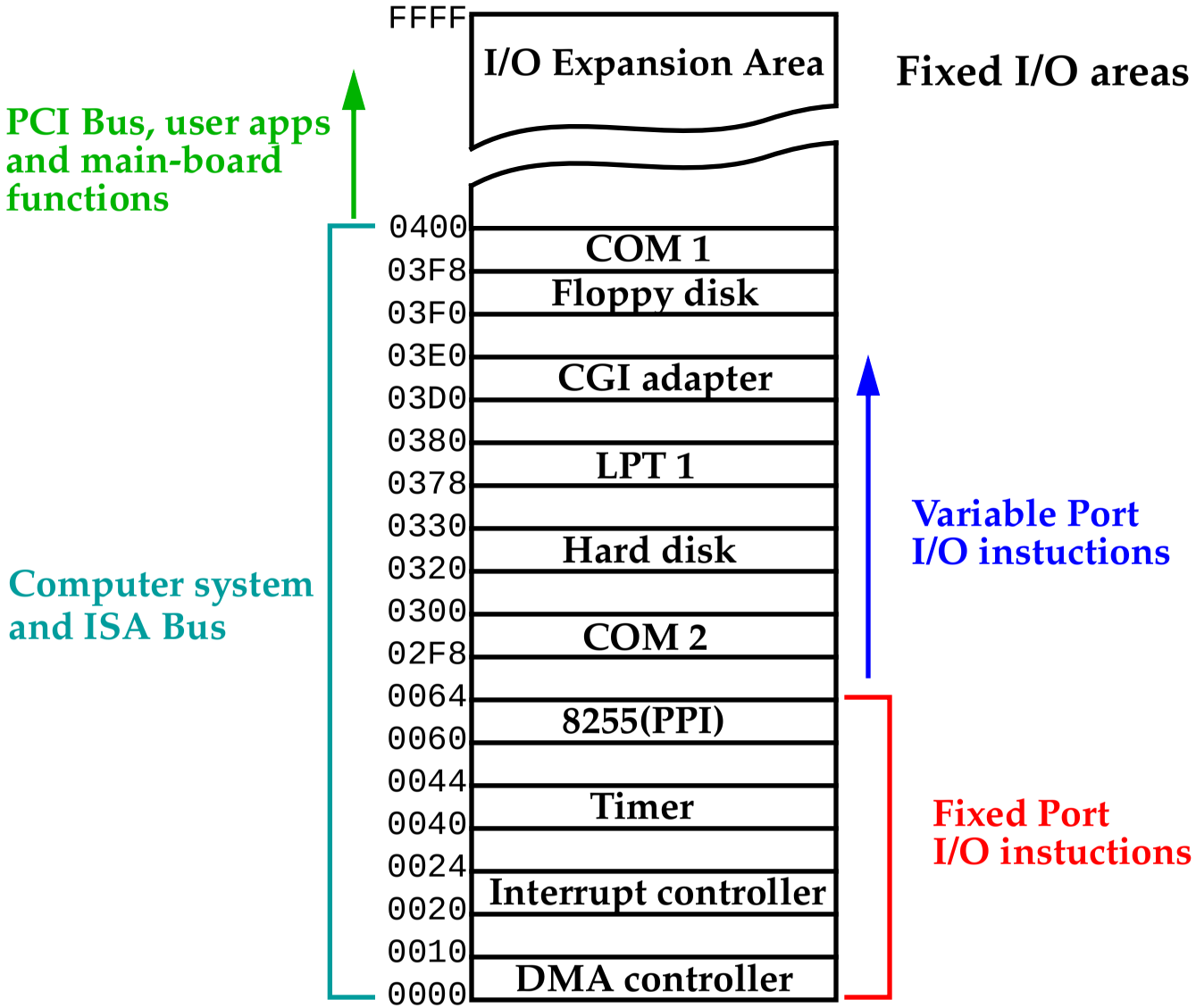
Advantage:
$\overline{IORC}$ and $\overline{IOWC}$ not required.
Any data transfer instruction.

Disadvantage:
Hardware using $M/\overline{IO}$ and $W/\overline{R}$ needed to develop signals $\overline{IORC}$ and $\overline{IOWC}$.
Requires IN, OUT, INS and OUTS

8-bit port addresses used to access system board device, e.g. timer and key-board.

16-bit port addresses used to access serial and parallel ports, harddrives, etc.

## I/O Map

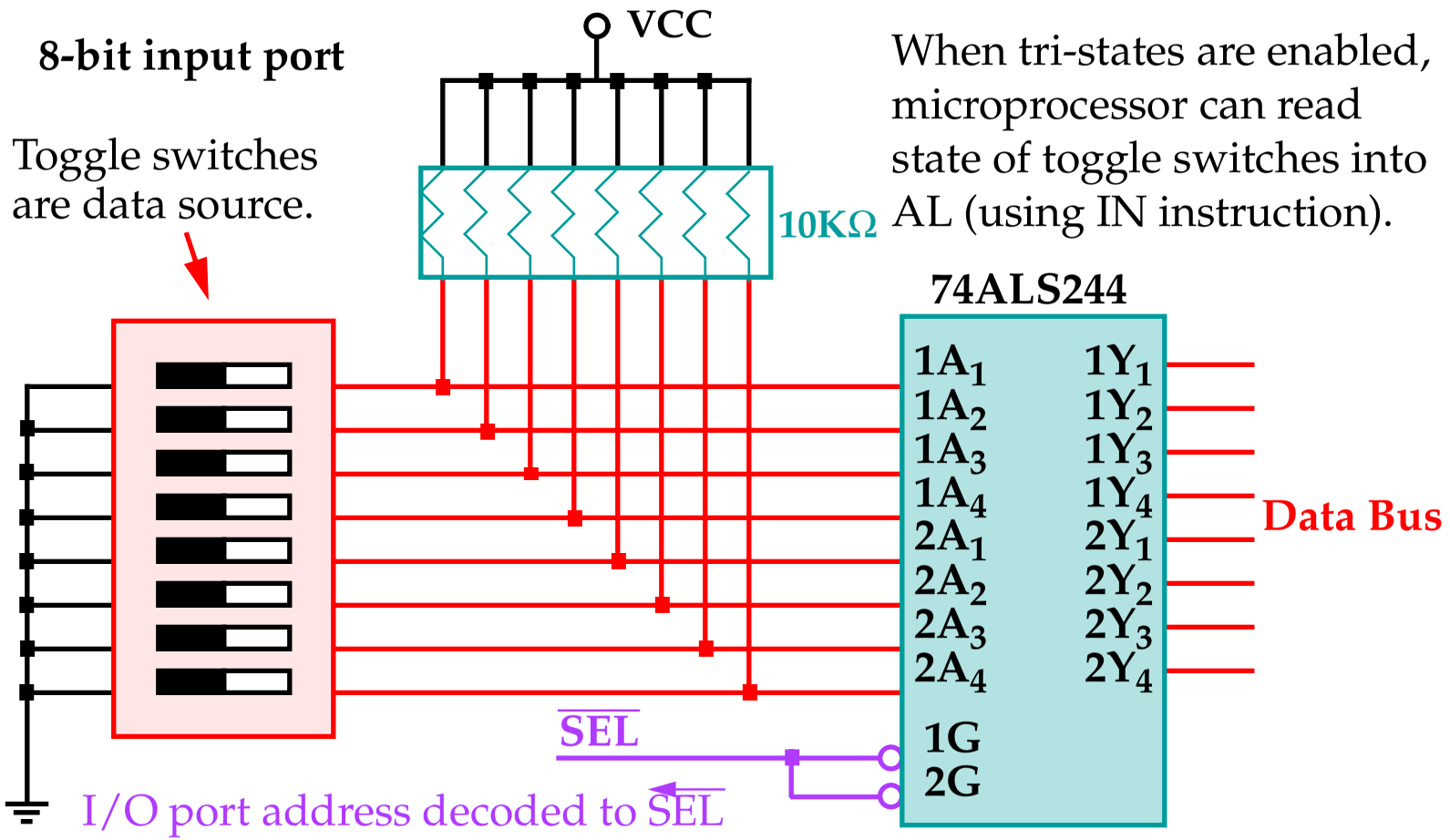

FFFF
I/O Expansion Area
Fixed I/O areas
PCI Bus, user apps and main-board functions
0400
COM 1
03F8
Floppy disk
03F0
03E0
CGI adapter
03D0
0380
LPT 1
0378
0330
Hard disk
0320
Computer system and ISA Bus
0300
COM 2
02F8
0064
8255(PPI)
0060
0044
Timer
0040
0024
Interrupt controller
0020
0010
DMA controller
0000
Variable Port I/O instuctions
Fixed Port I/O instuctions

## Basic I/O Interface

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

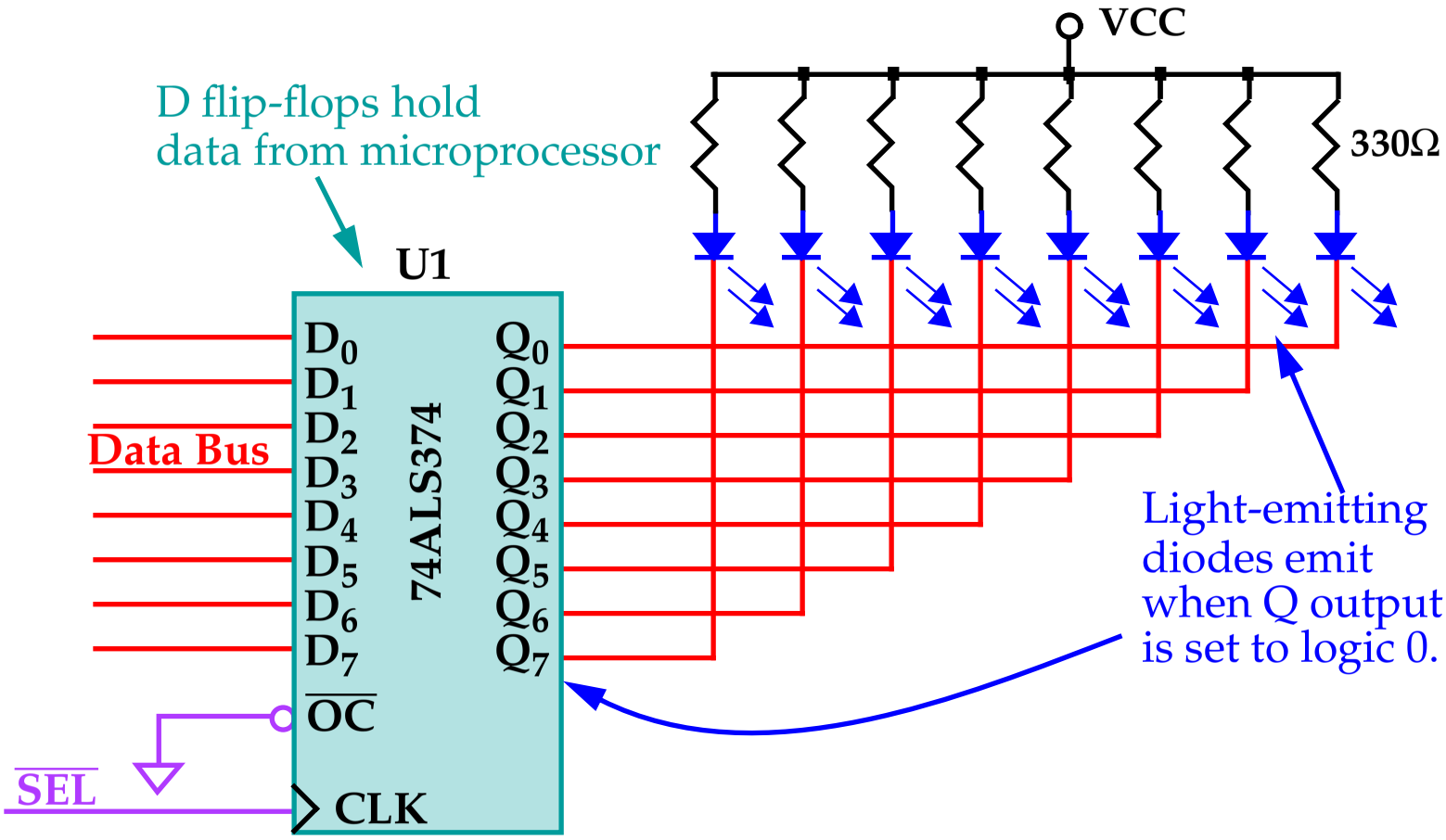
The basic output device (from the microprocessor) is a set of latches.

Basic Input Interface:



## Basic I/O Interface

Basic Output Interface:




In this case, the data from the OUT instruction is latched using $\overline{\text{SEL}}$.

**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.

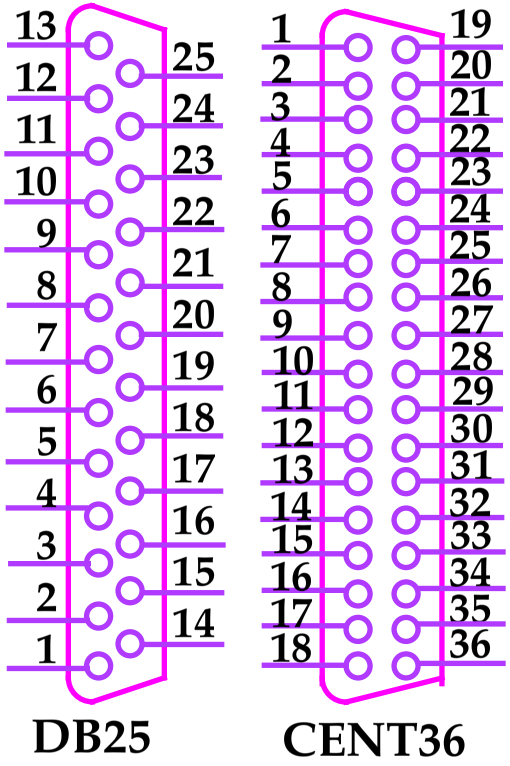
There are two basic mechanisms for the processor to service a device.

- 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.

**Handshaking**

A printer connected to the parallel port requires handshaking.

The parallel port specification is shown below:




| DB25 | CENT36 | Function | DB25 | CENT36 | Function |
|---|---|---|---|---|---|
| 1 | 1 | Data Strobe | 12 | 12 | Paper out |
| 2 | 2 | Data0 | 13 | 13 | Select |
| 3 | 3 | Data1 | 14 | 14 | Afd |
| 4 | 4 | Data2 | 15 | 32 | $\overline{\text{Error}}$ |
| 5 | 5 | Data3 | 16 | - | $\overline{\text{RESET}}$ |
| 6 | 6 | Data4 | 17 | 31 | Select in |
| 7 | 7 | Data5 | 18-25 | 19-30 | GND |
| 8 | 8 | Data6 | - | 17 | Frame GND |
| 9 | 9 | Data7 | - | 16 | GND |
| 10 | 10 | $\overline{\text{Ack}}$ | - | 33 | GND |
| 11 | 11 | Busy | | | |

The processor writes ASCII data out to the Datax pins of the printer and toggles the Data Strobe pin to latch it in.

The printer raises the Busy pin.

When the Busy pin goes low, the sequence repeats.

**Interfacing Circuitry**

The terminal characteristics of the processor must be matched to those of the I/O devices.

Input Devices:

They are either:

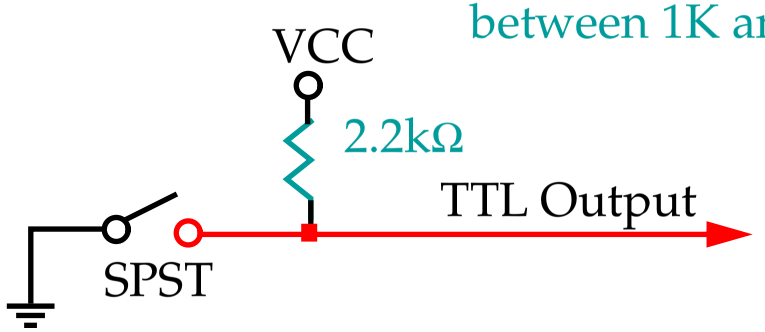
- TTL (0.0V-0.8V low and 2.0-5.0V high) or compatible.
- Switch-based; usually either open or connected.

These must be conditioned before they can be used properly.

For example, to make a simple (single-pole, single-throw) toggle switch TTL compatible:

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

This ensures that the output is held at either 0 or logic 1 at all times (it never floats).



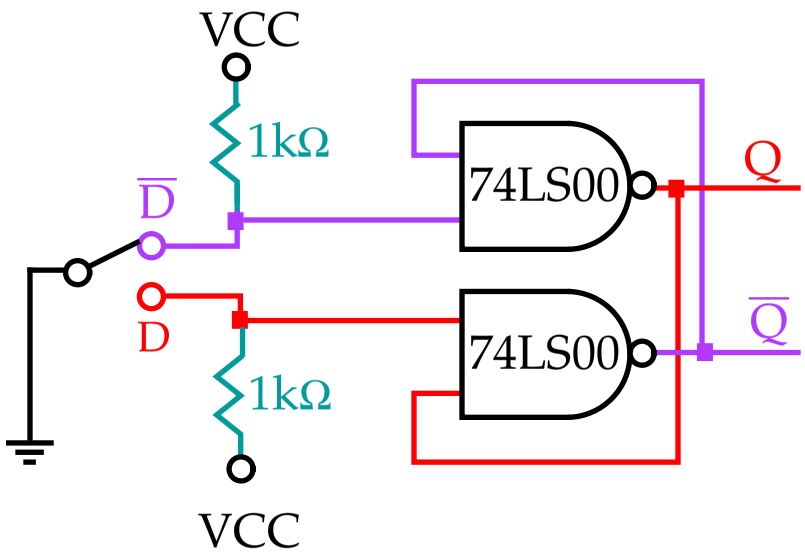
**Interfacing Circuitry**

Input Devices:

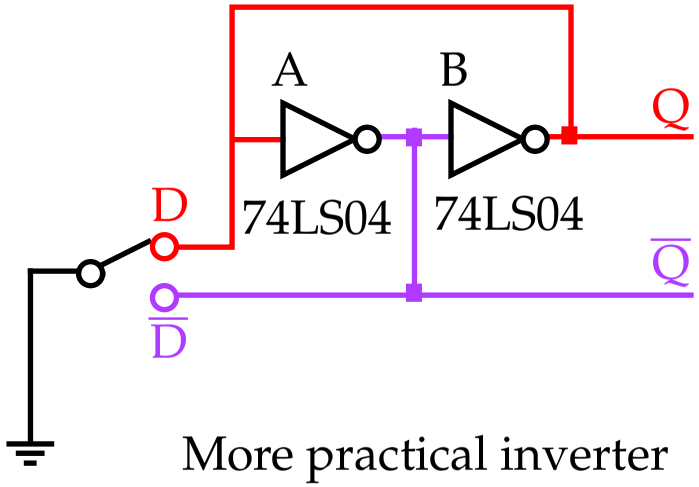
Mechanical switches physically bounce when they are closed (causing them to momentarily open after being closed).

This can cause a problem if they are used as a clocking signal.

Two asynchronous flip-flop solutions are given below:




Cross-coupled NANDs.




More practical inverter implementation.

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

**Interfacing Circuitry**

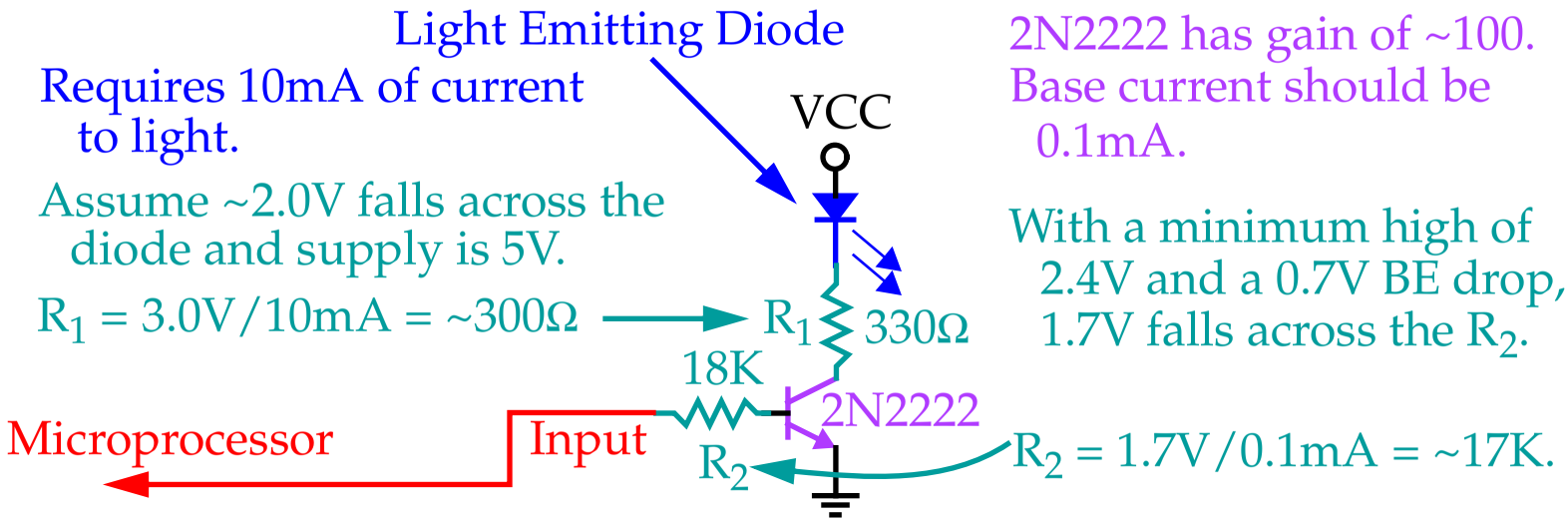
Output Devices:

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:



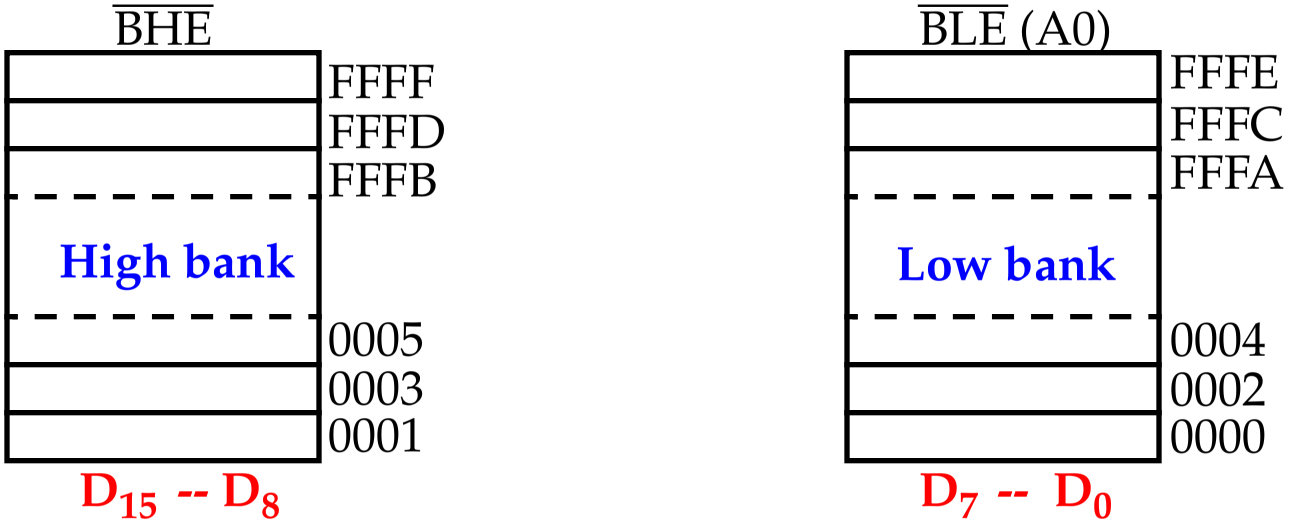
**I/O Port Decoding**

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/$\overline{\text{R}}$ pins of the microprocessor.

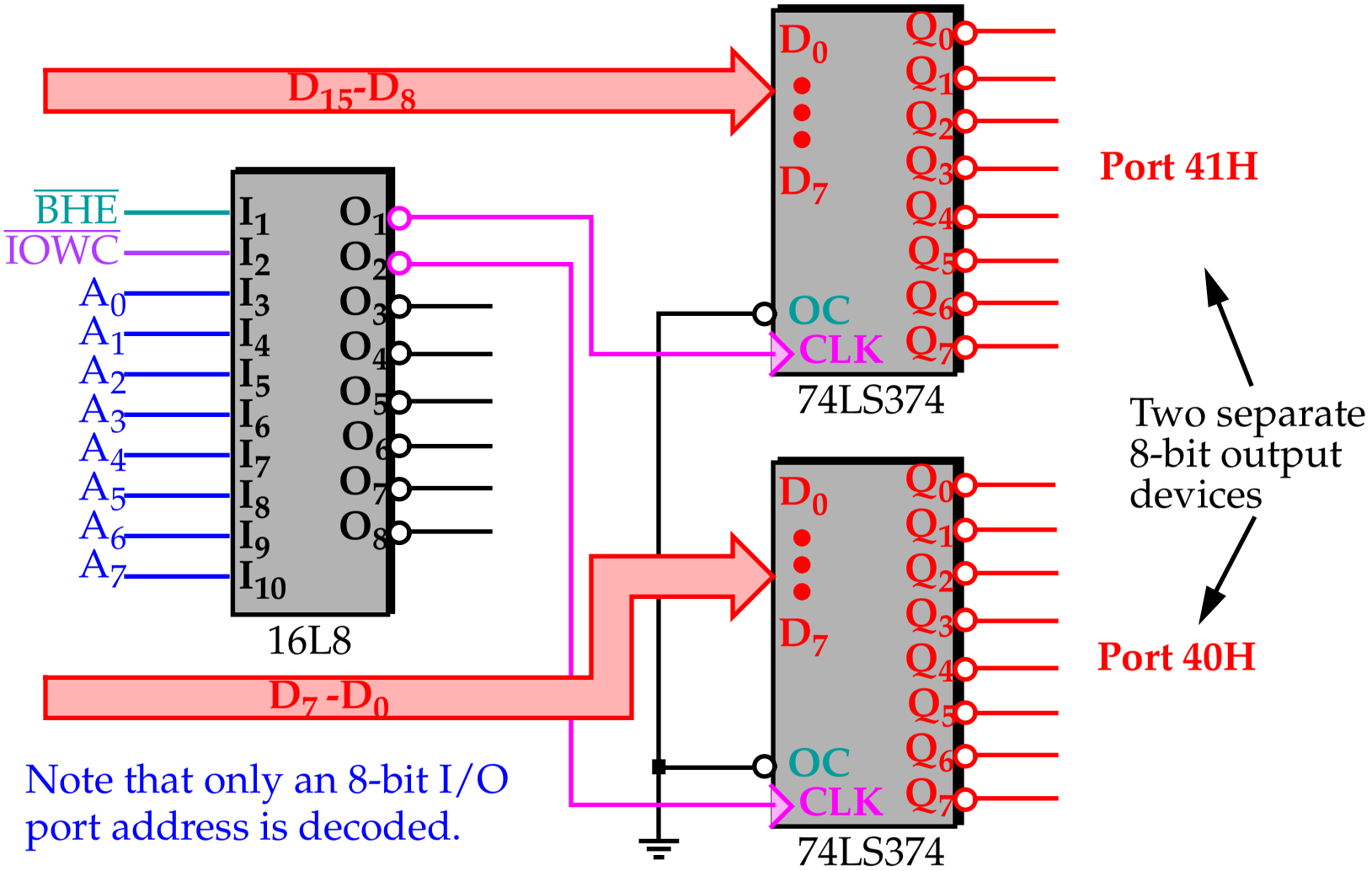
The text gives examples of 8-bit decoding and 16-bit decoding, which is a straightforward application of devices we've used for memory decoding.

The I/O banks on the 8086 through the 80386SX are also set up like the memory.



## 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 $\overline{\text{BHE}}$ 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.