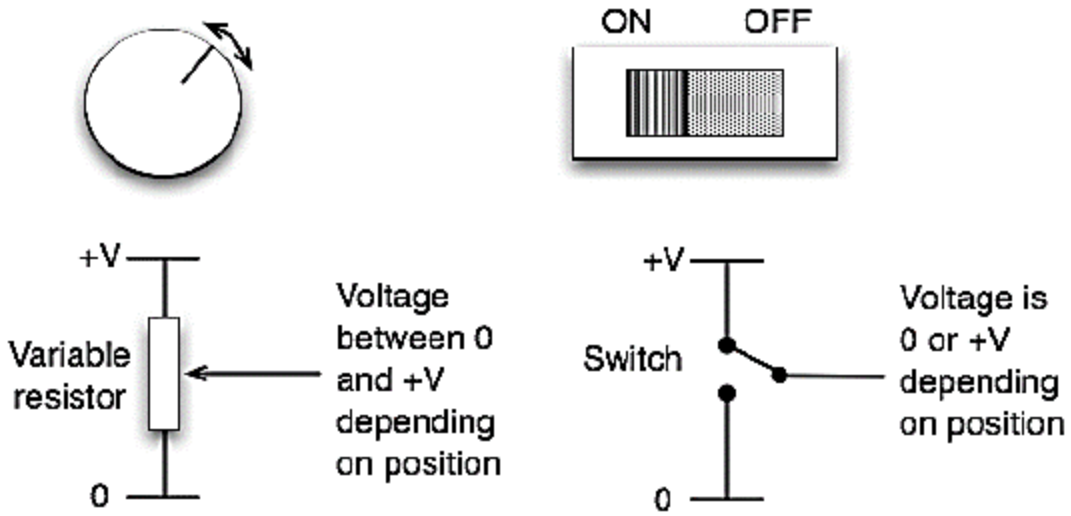
Basic interfacing Devices
Un – Programmable interfacing devices

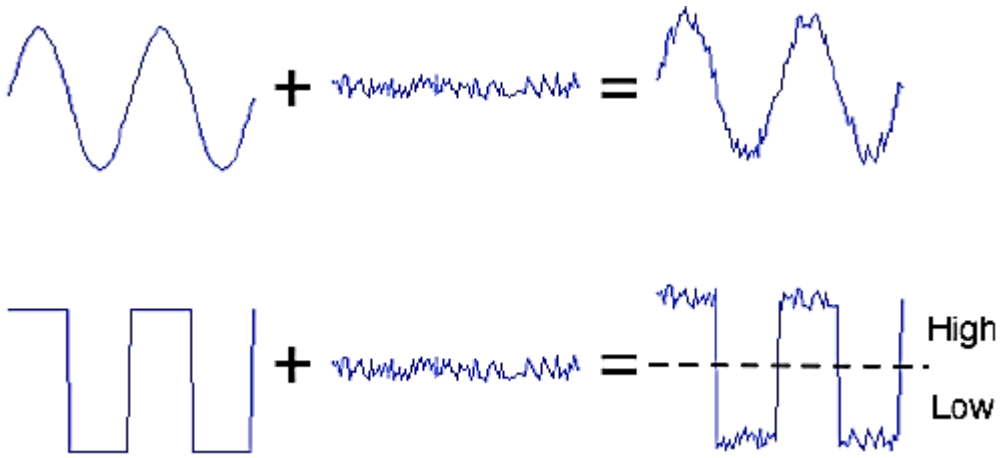
Introduction to digital system

Analogue vs Digital

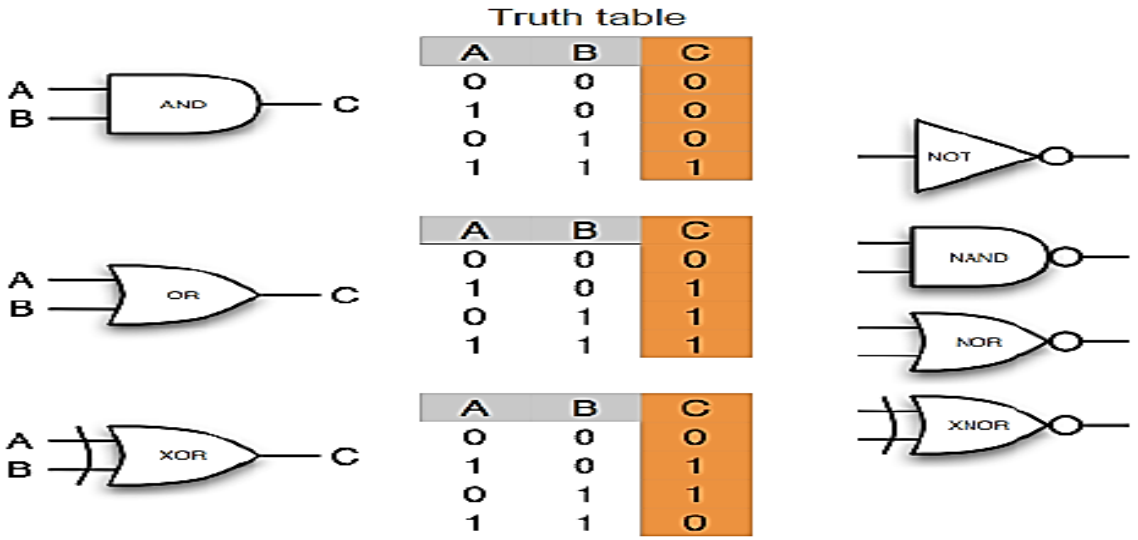
- Analog information is made up of a continuum of values within a given range
- The term digital refers to the fact that the signal is limited to only a few possible values. In general, digits signals are represented by only two possible voltages on a wire - 0 volts (which we called "binary 0", or just "0") and 5 volts (which we call "binary 1", or just "1"). We sometimes call these values "low" and "high", or "false" and "true".



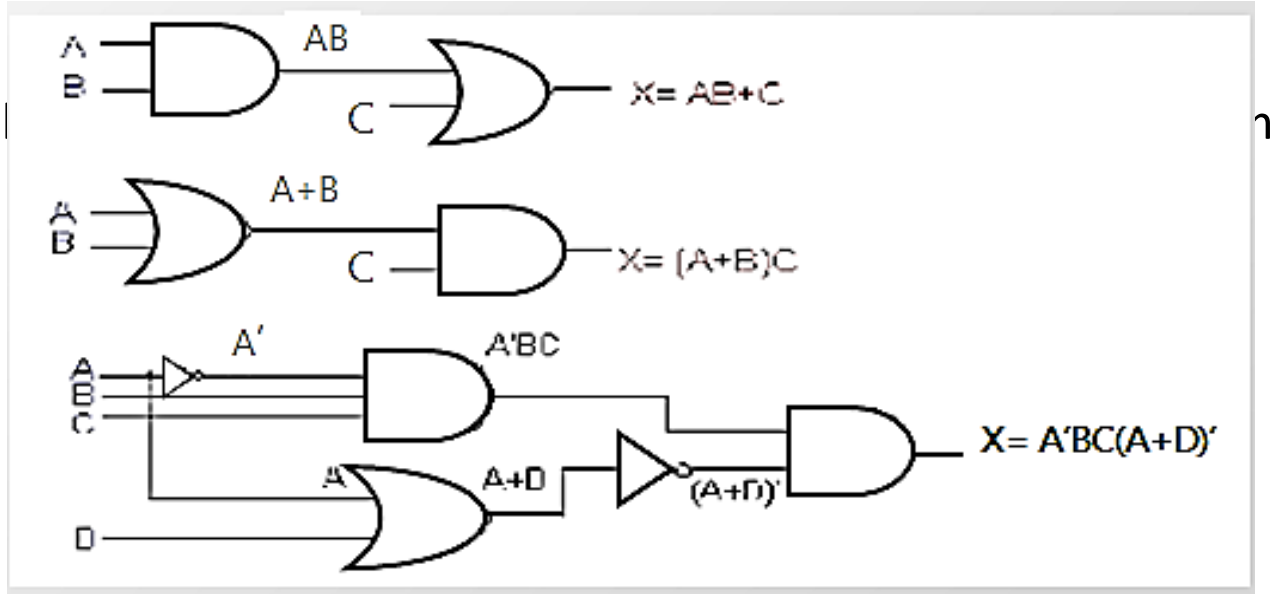
- Digital Information is less susceptible to noise than analog information
- Exact voltage values are not important, only their class (1 or 0)
- The complexity of operations is reduced, thus it is easier to implement them with high accuracy in digital form
- BUT: Most physical quantities are analog, thus a conversion is needed



- **Logic operations**

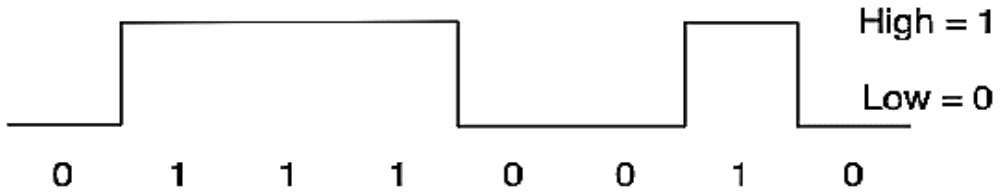


- These l ns.



Binary system

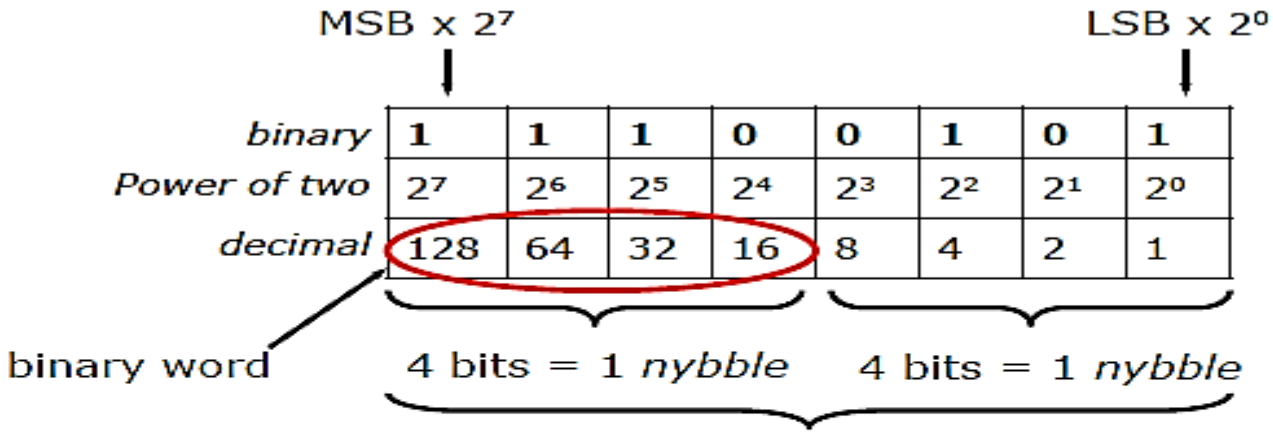
- Digital systems represent information using a binary system, where data can assume one of only two possible values: zero or one (Low or High).



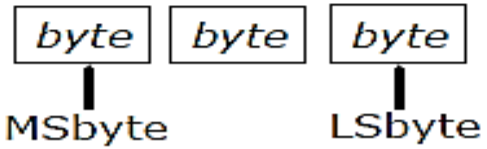
- The binary system represents numbers using binary digits (bits) where each digit corresponds to a power of two.

binary	1	1	1	0	0	1	0	1
Power of two	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
decimal	128	64	32	16	8	4	2	1

- The total (in decimal) is 128 + 64 + 32 + 4 + 1 =229
- Since we begin counting from zero, N bits can represent 2^N values: from 0 to 2^{N-1} inclusive (e.g. 256 values, from 0 to 255, for 8 bits).
- Groups of bits form binary words .



Unit	Value
Kilobyte (KB)	1024 Bytes
Megabyte (MB)	1024 KBytes
Gigabyte (GB)	1024 MBytes
Terabyte (TB)	1024 GBytes



- How to convert from decimal to binary?
- Repeat division by 2
- Example: Convert 29_{10} to binary
  - 29/2 = 14 remainder 1 (LSB)
  - 14/2 = 7 remainder 0
  - 7/2 = 3 remainder 1
  - 3/2 = 1 remainder 1
  - 1/2 = 0 remainder 1 (MSB)
- 29_{10} => 11101_2

Hexadecimal numbers

- The hexadecimal number system (AKA hex) is a base 16 notation. It is the most popular large-base system for representing binary numbers.
- Values in MIDI implementation charts are often expressed as hexadecimal numbers.
- Each symbol represents 4-bits (1 nybble), that can take one of 16 different values: the values 0-9 are represented by the digits 0-9, and the values 10-15 are represented by the capital letters A-F.
- Conversions are performed as with the other number systems.

Bin	Hex	Dec	Bin	Hex	Dec	Bin	Hex	Dec	Bin	Hex	Dec
0000	0	0	0100	4	4	1000	8	8	1100	C	12
0001	1	1	0101	5	5	1001	9	9	1101	D	13
0010	2	2	0110	6	6	1010	A	10	1110	E	14
0011	3	3	0111	7	7	1011	B	11	1111	F	15

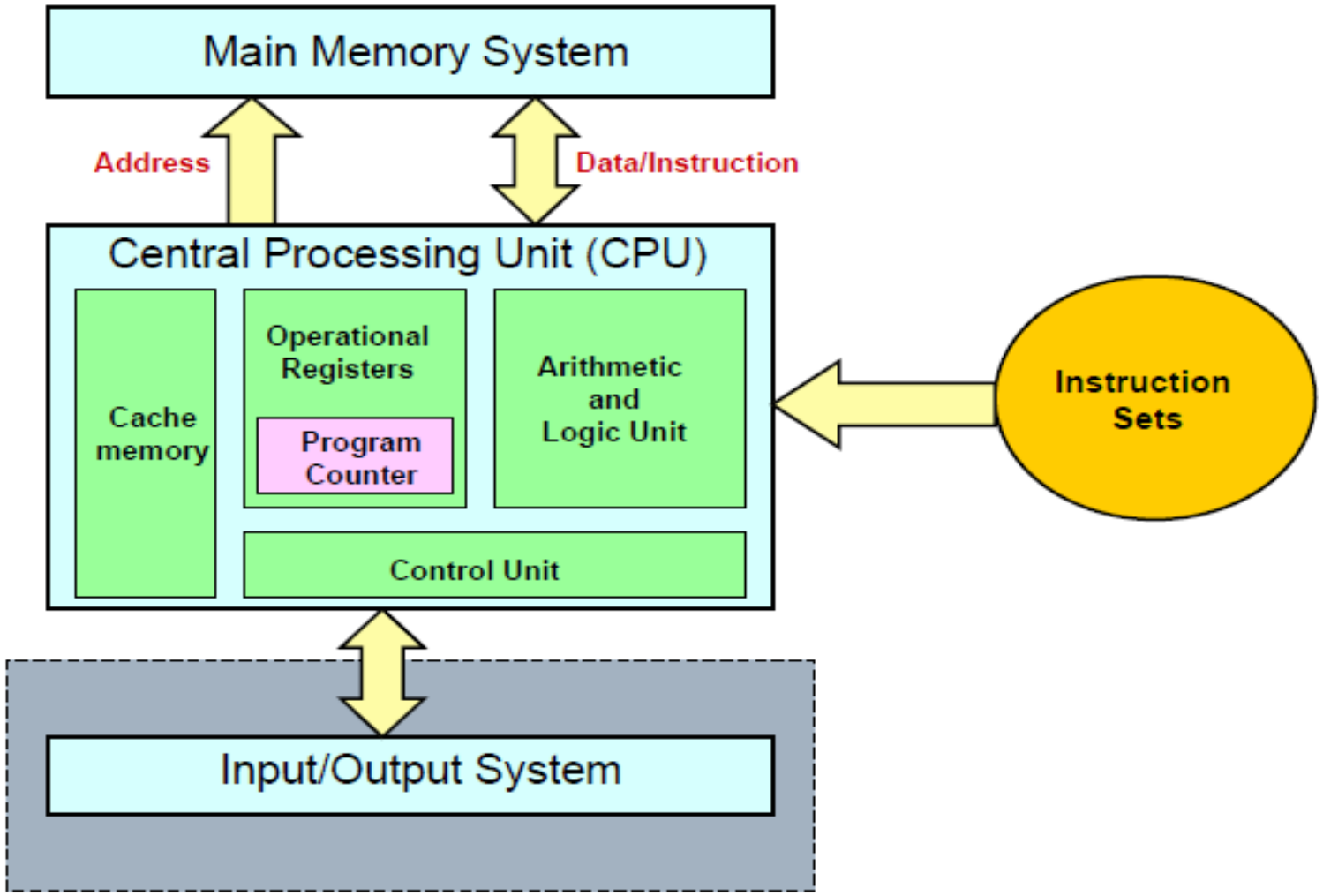
Basic of interface devices (IDF)

Any application of microprocessor based system required the **transfer of data between external circuitry to the microprocessor and verse versa.** Hence interfacing is used for this purpose to exchange information between two different applications/devices.

An interface device (IDF) is a hardware component or system of components that allows a human being to interact with a computer, digital system , or any other electronic information system.

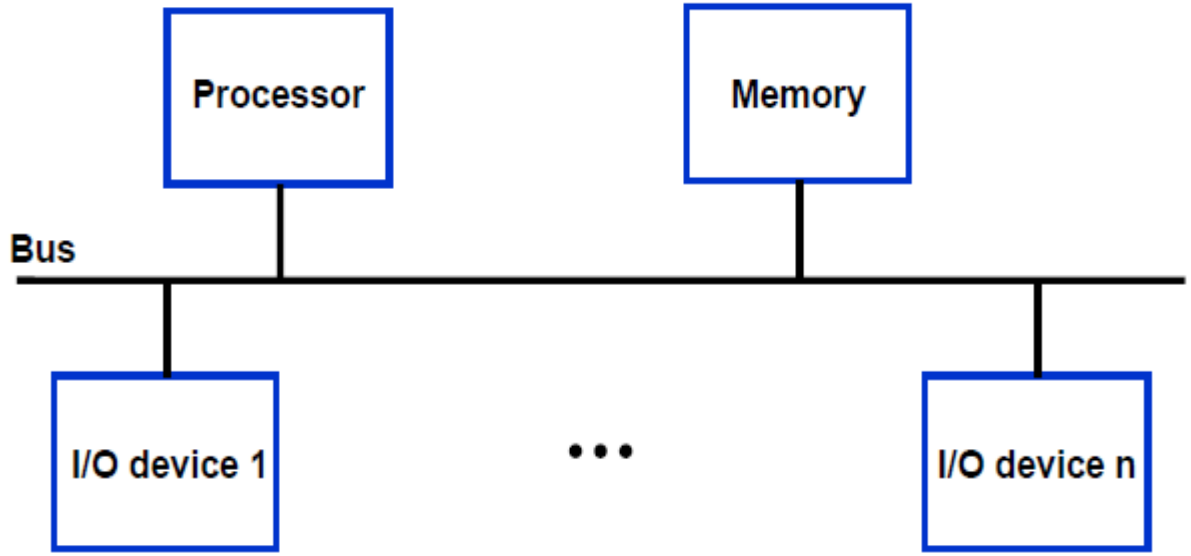
- An interface is a device and/or set of rules to match the output of one device to send information's to the input of another device
- The process of connecting devices together so that they can exchange information's.
- The process of reading input signals and sending output signals is called I/O.

Main components of digital system



Tools for information exchange

- **Accessing I/O Devices**
- Single-bus structure
- The bus enables all the devices connected to it to exchange information.
- Typically, the bus consists of three sets of lines used to carry address, data, and control signals
- Each I/O device is assigned a unique set of addresses



I/O Interface for an Input Device

The address decoder, the data and status registers, and the control circuitry required to coordinate I/O transfers constitute the device's ***interface circuit***

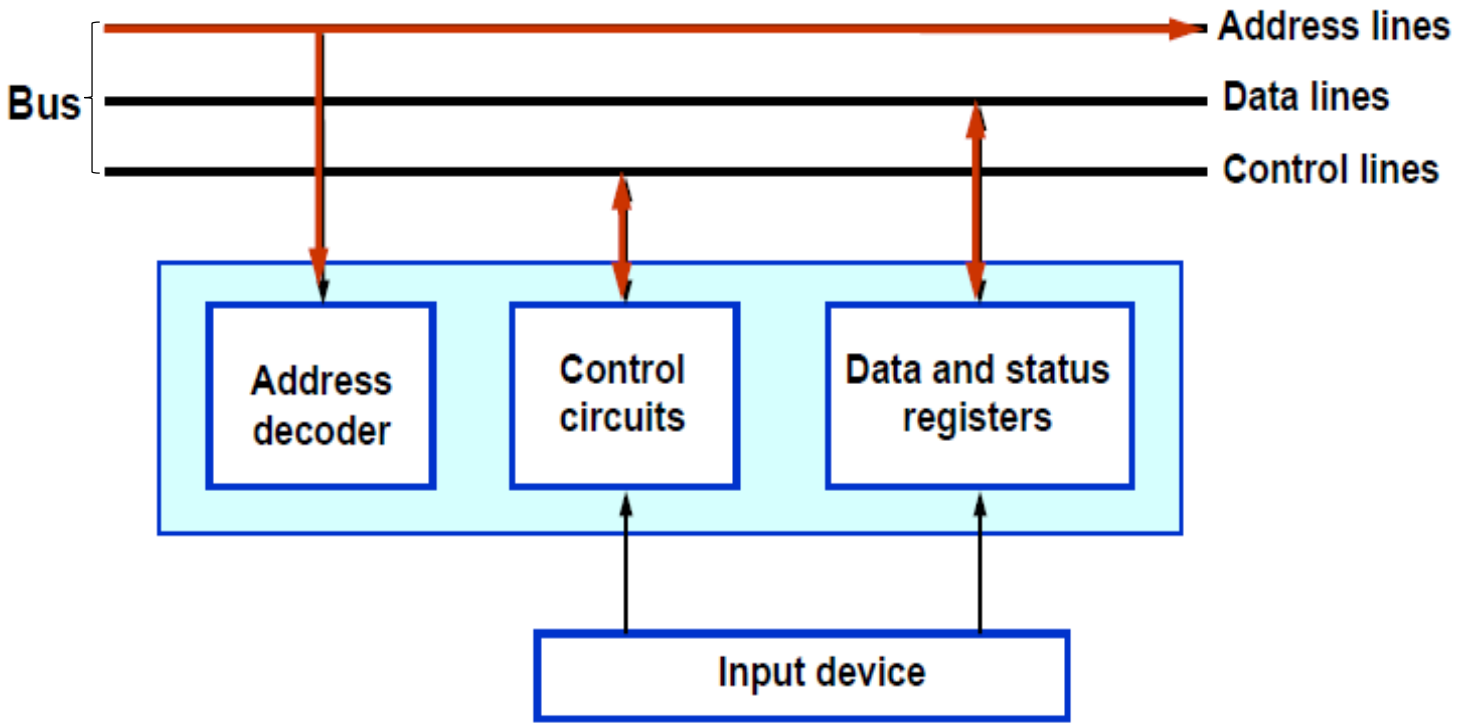


Figure 1. block diagram for input device interface

I/O Techniques (Interfacing techniques)

- The unprogrammable interfacing.
- The programmable interfacing.
- Interrupt (interrupt controller 8259)
- Direct Memory Access (DMA)

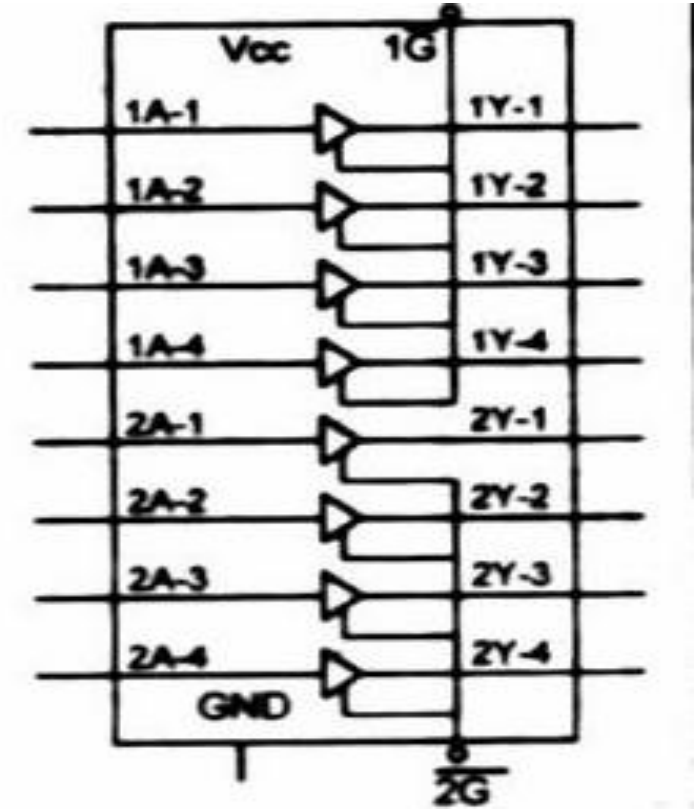
The Un - Programmable Interfacing

The basic input device is a set of three-state buffers. The basic output device is a set of data latches. The term **IN** refers to moving data from the I/O device into the microprocessor and the term **OUT** refers to moving data out of the digital system to the I/O device. We'll see the design of simple I/O ports using TTL logic gates 74LS373 and 74LS244.

The Basic Input Interface.

When data is coming in by way of a data bus, it must come in through a three-state buffer. This is referred to as tristated, which comes from the term tri-state buffer.

- A tri-state buffer is internal and therefore invisible.
- For the simple input ports we use the 74LS244 chip. See the figure beside.
- Notice that since 1G and 2G each control only 4 bits of the 74LS244, they both must be activated for the 8-bit input.
- The gates must controlled (connected) to decoder according to address selected.



74LS244 Octal Buffer

Analyze Figures 11-5 and 11-6

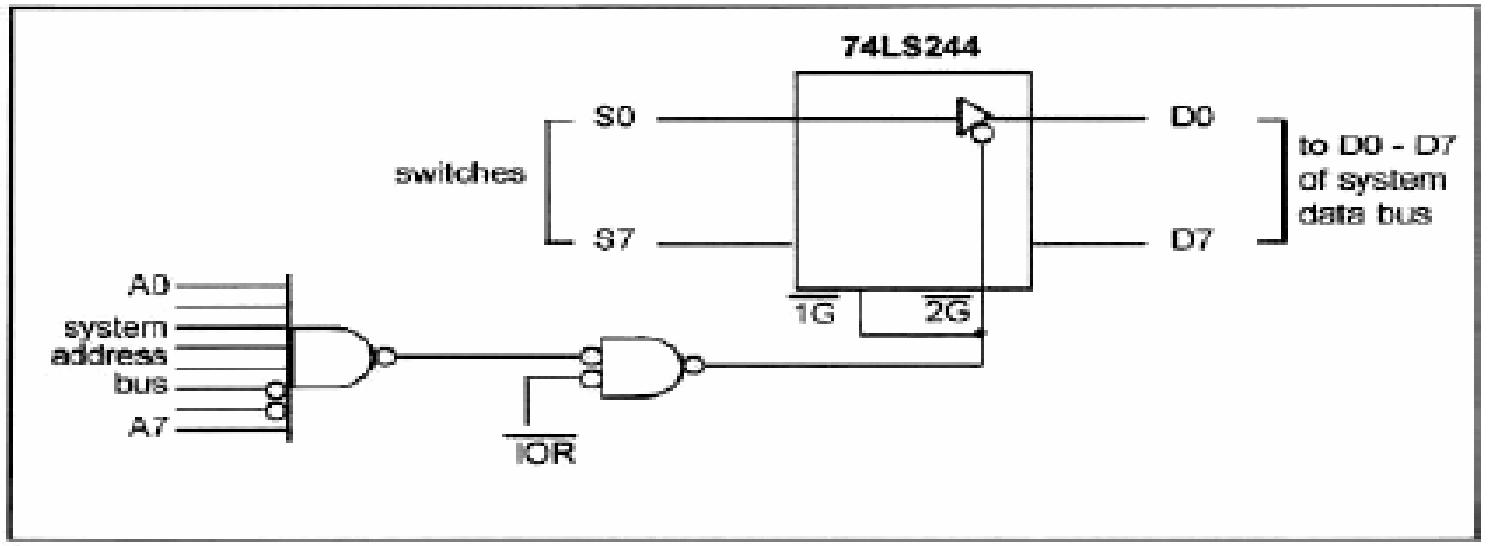


Figure 11-5. Design for "IN AL,9FH"

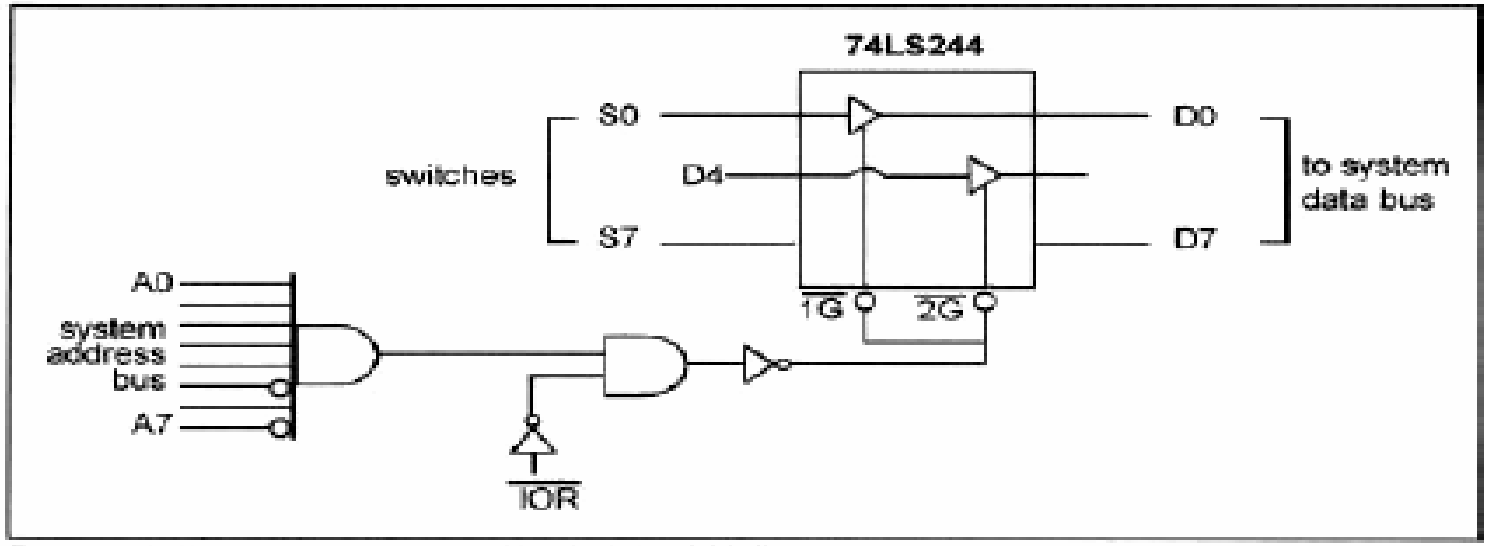


Figure 11-6. Input port design for "IN AL,5FH"

Problem. What change(s) would you do to change the design to “IN AL,6AH”?

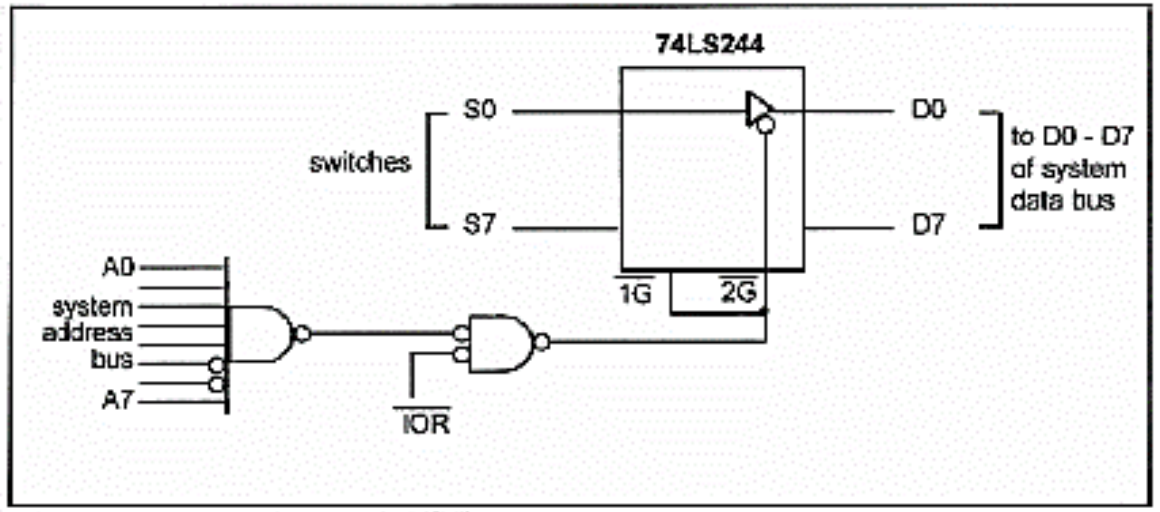
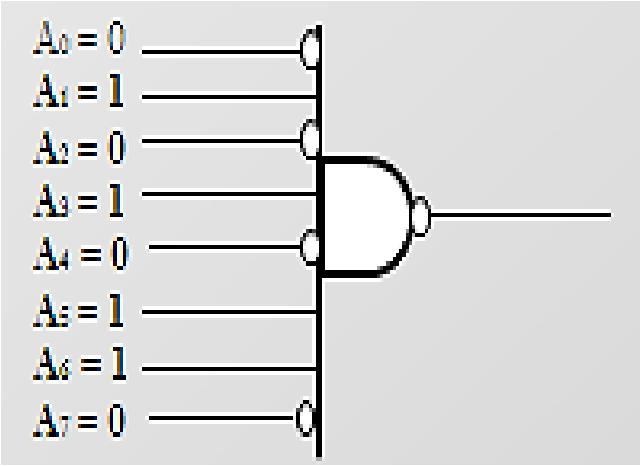


Figure 11-5. Design for "IN AL,8FH"

Solution: The decoder must be design for 6AH as follows :

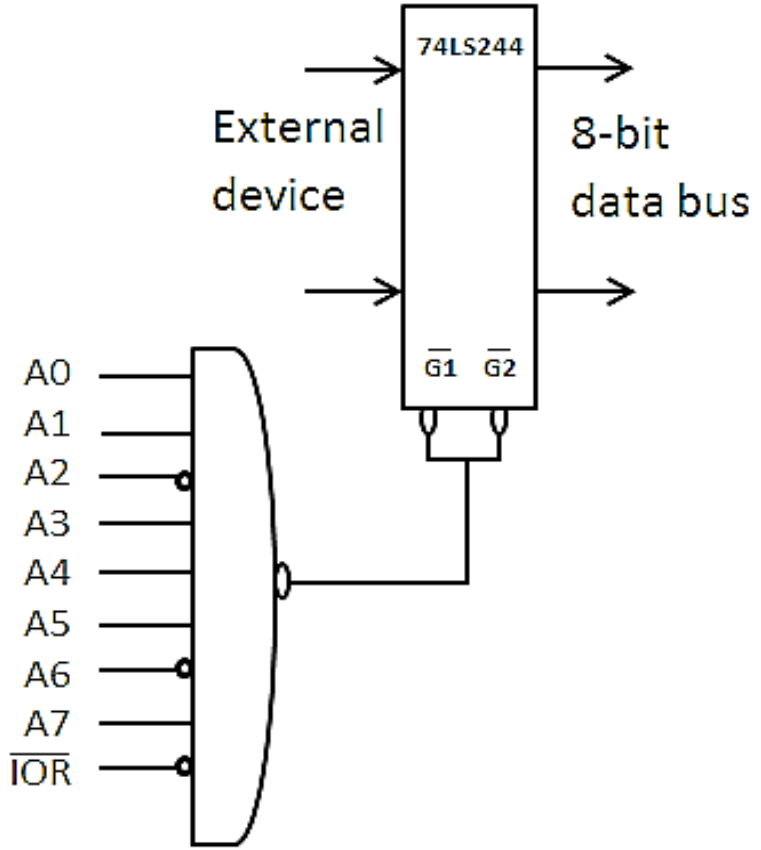


Example // Draw the input interface circuit, and explain the port address selection when execute the command **IN AL,BBH**

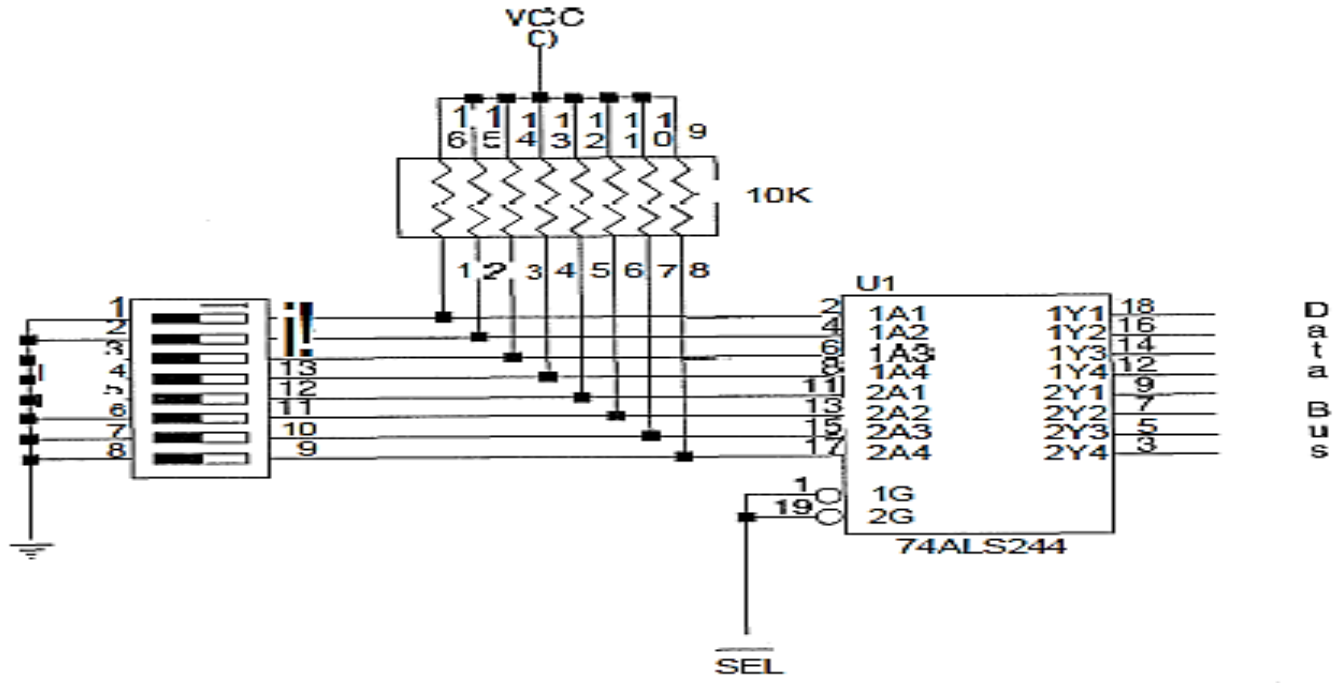
SOLUTION:

BBH = 1 0 1 1 1 0 1 1 Binary

A7 A6 A5 A4 A3 A2 A1 A0



Three-state buffers are used to construct the 8-bit input port in Figure below. The external TTL data (simple toggle switches in this example) are connected to the inputs of the buffers. The outputs of the buffers connect to the data bus. Circuit of Figure 11-3 allows the digital system to read the contents of the 8 switches that connect to any 8-bit section of the data bus when the select signal SEL becomes a logic 0.

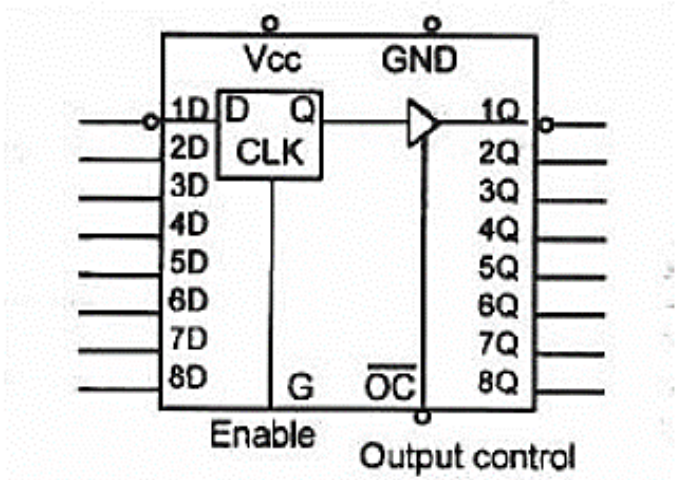


The basic input interface of 74LS244.

This basic input circuit is not optional and must appear any time that input data are interfaced to the microprocessor.

The Basic Output Interface.

The basic output interface receives data from the microprocessor and must usually hold it for some external device. Its latches or flip-flops, like the buffers found in the input device, are often built into the I/O device. The 74LS373 as an output port used for this reason. In every computer, whenever data is sent out by the CPU via the data bus, the data must be latched by the receiving device the 74LS373 can be used for this purpose.



Function Table

Output Control	Enable G	D	Output
L	H	H	H
L	H	L	L
L	L	X	Q0
H	X	X	Z

Analyze Figure (a). Under what condition(s) 74LS373 is activated?

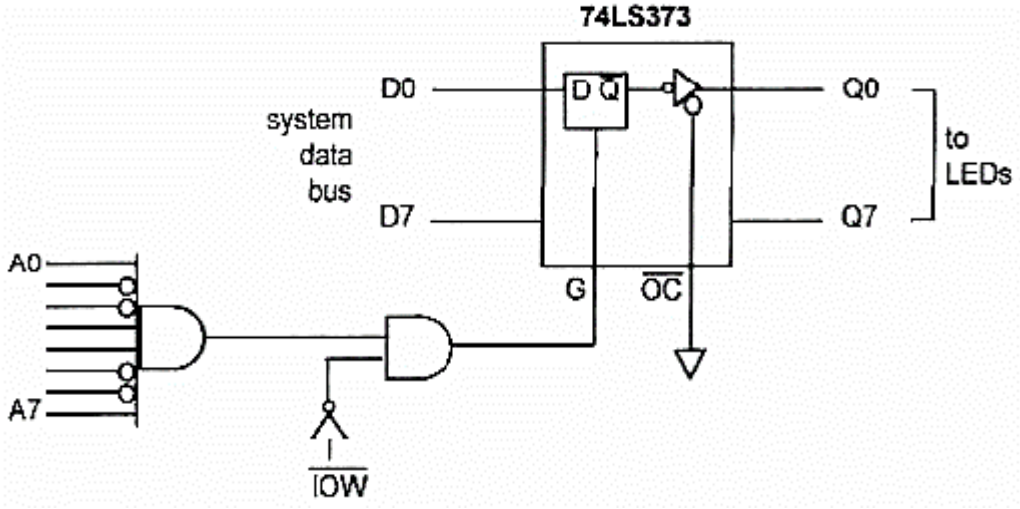


Figure (a). Design for “OUT 99H,AL”

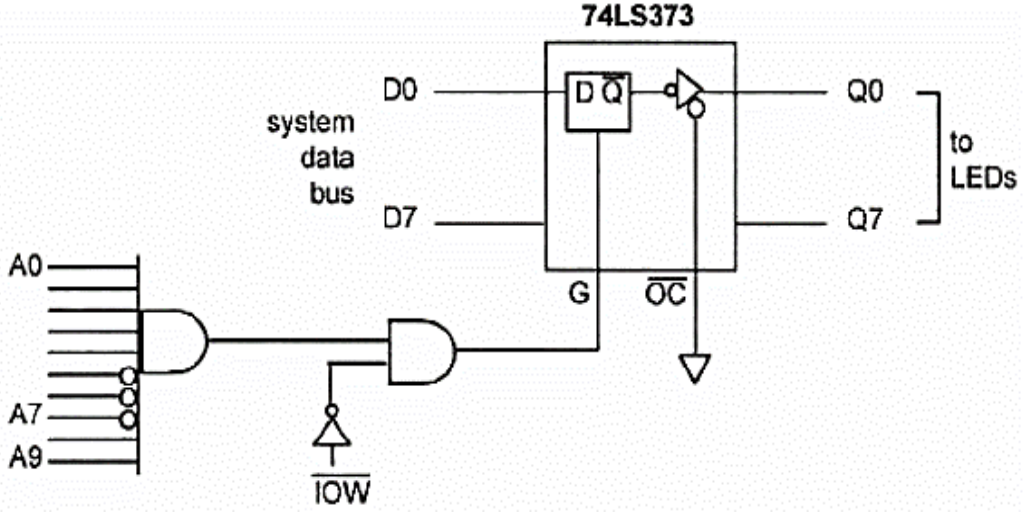


Figure (b). Design for Output port address of 31FH

Example // Draw the output interface circuit, and explain the port address selection when execute the command **OUT C5H,AL**

SOLUTION:

BBH = 1 1 0 0 0 1 0 1 Binary

A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0

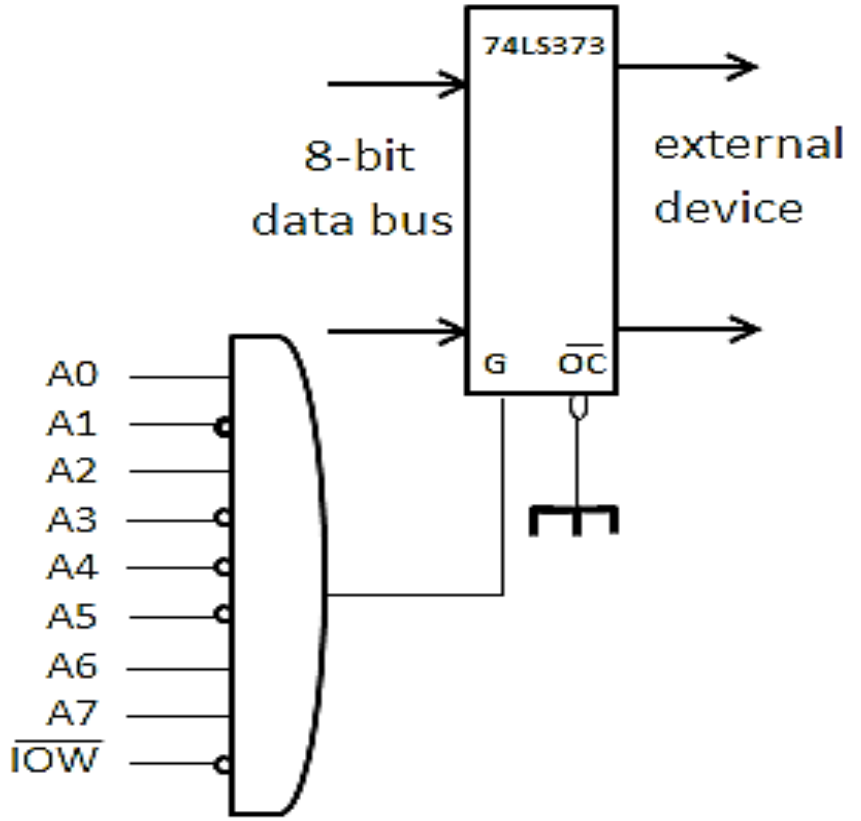
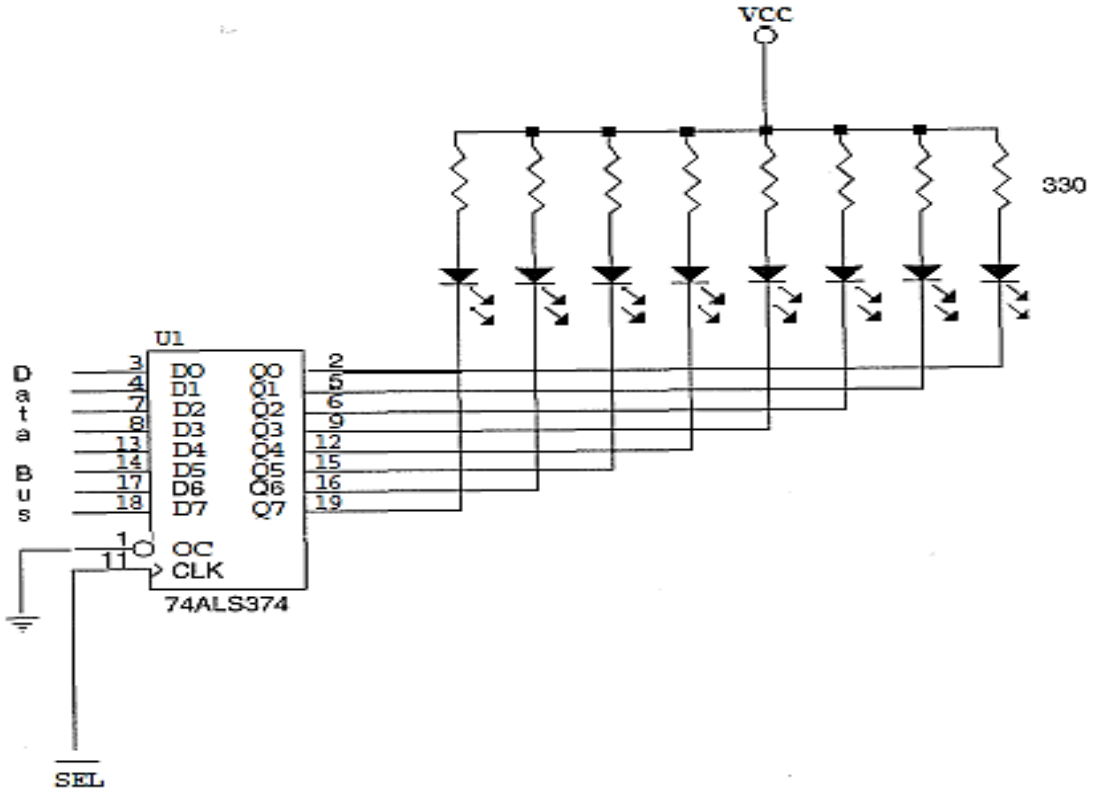


Figure below shows how eight simple light-emitting diodes (LEDs) connect to the microprocessor through a set of eight data latches. The latch stores the number output by the microprocessor from the data bus so that the LEDs can be lit with any 8-bit binary number. Latches are needed to hold the data because when the microprocessor executes an OUT instruction, the data are only present on the data bus for less than 1.0 µS. Without a latch, the viewer would never see the LEDs illuminate.

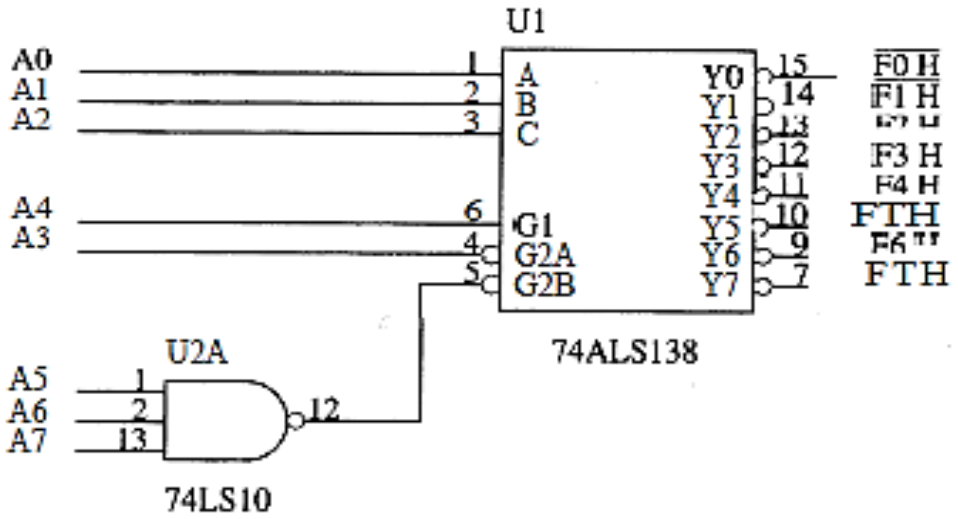


The basic output interface connected to a set of LED displays.

Decoding 8-Bit I/O Addresses

As mentioned, the fixed I/O instruction uses an 8-bit I/O port address that appears on A15-AO as OOOOH-OOFFH. If a system will never contain more than 256 I/O devices, we often decode only address connections A7-AO for an 8-bit I/O port address and the upper address A15-A8 will be ignore . Embedded systems often use 8-bit port addresses. If the address is decoded as an 8-bit address, we can never include I/O devices that use a 16-bit I/O address.

Figure below illustrates a 74ALS138 decoder that decodes 8-bit I/O ports FOH through F7H. (We assume that this system will only use I/O ports OOH-FFH for this decoder example.) This decoder is identical to a memory address decoder except we only connect address bits A7-AO to the inputs of the decoder.



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

Handshaking

Many I/O devices accept or release information at a much slower rate than the microprocessor. Another method of I/O control, called handshaking or polling, synchronizes the I/O device with the microprocessor. An example device that requires handshaking is a parallel printer that prints 100 characters per second (CPS). It is obvious that the microprocessor can send more than 100 CPS to the printer, so a way to slow the microprocessor down to match speeds with the printer must be developed.