UNIT III

Combinational logic circuits: Multiplexers – Demultiplexers –Decoders -Encoders.

Multiplexer and Demultiplexer:

A multiplexer is a circuit that accept many input but give only one output. A demultiplexer function exactly in the reverse of a multiplexer, that is a demultiplexer accepts only one input and gives many outputs. Generally multiplexer and demultiplexer are used together, because of the communication systems are bi directional.

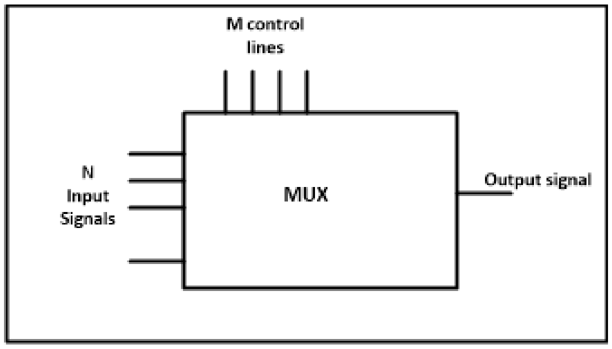
Multiplexer means many into one. A multiplexer is a circuit used to select and route any one of the several input signals to a signal output. An simple example of an non electronic circuit of a multiplexer is a single pole multiposition switch.

Multiposition switches are widely used in many electronics circuits. However circuits that operate at high speed require the multiplexer to be automatically selected. A mechanical switch cannot perform this task satisfactorily. Therefore, multiplexer used to perform high speed switching are constructed of electronic components.

Multiplexer handle two type of data that is analog and digital. For analog application, multiplexer are built of relays and transistor switches. For digital application, they are built from standard logic gates.

The multiplexer used for digital applications, also called digital multiplexer, is a circuit with many input but only one output. By applying control signals, we can steer any input to the output. Few types of multiplexer are 2-to-1, 4-to-1, 8-to-1, 16-to-1 multiplexer.

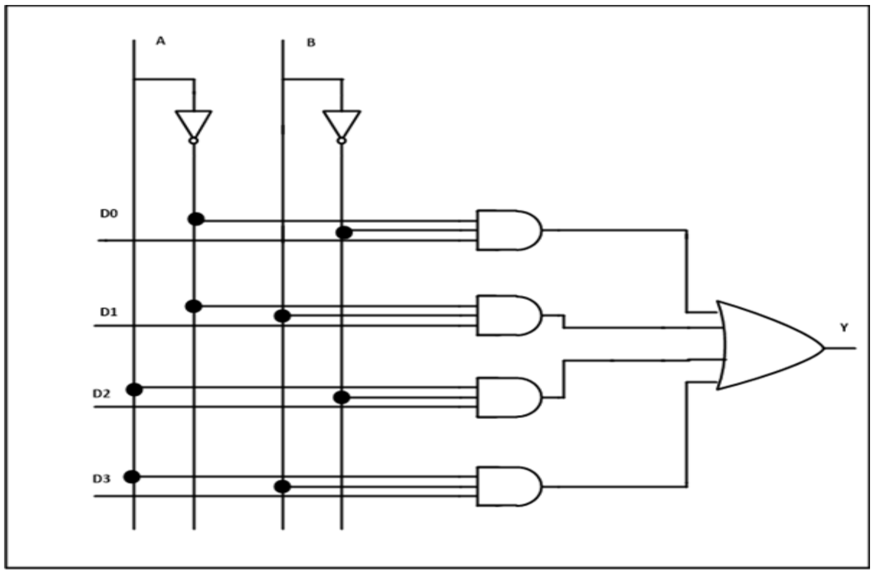
Following figure shows the general idea of a multiplexer with n input signal, m control signals and one output signal.



Multiplexer Pin Diagram

Understanding 4-to-1 Multiplexer:

The 4-to-1 multiplexer has 4 input bit, 2 control bits, and 1 output bit. The four input bits are D0,D1,D2 and D3. only one of this is transmitted to the output y. The output depends on the value of AB which is the control input. The control input determines which of the input data bit is transmitted to the output.
For instance, as shown in fig. when AB = 00, the upper AND gate is enabled while all other AND gates are disabled. Therefore, data bit D0 is transmitted to the output, giving Y = D0.

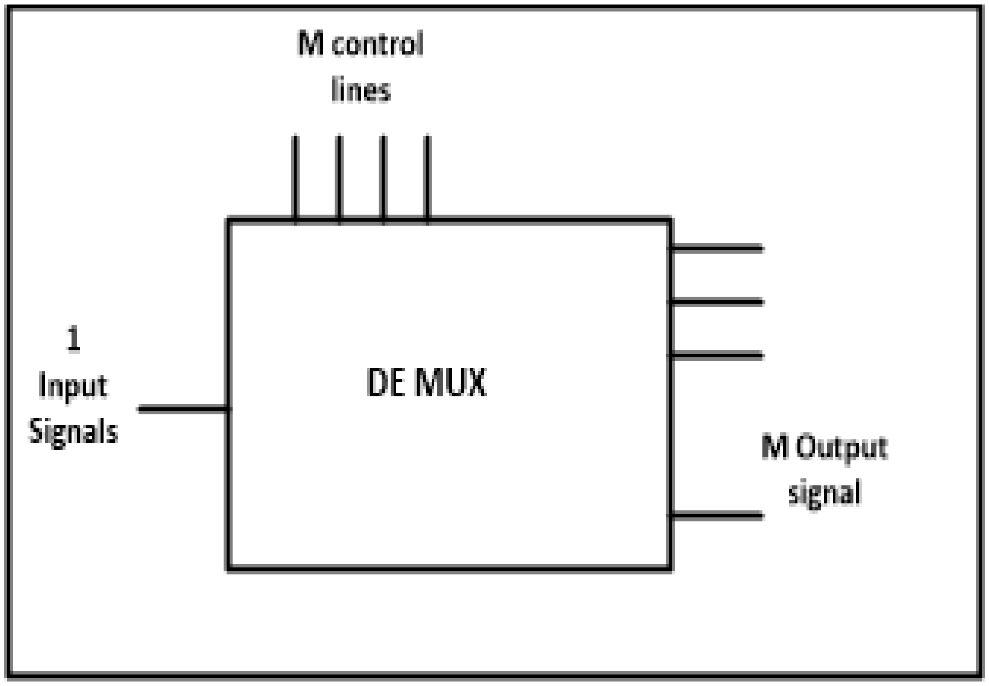


4 to 1 Multiplexer Circuit Diagram

If the control input is changed to AB =11, all gates are disabled except the bottom AND gate. In this case, D3 is transmitted to the output and Y = D3.

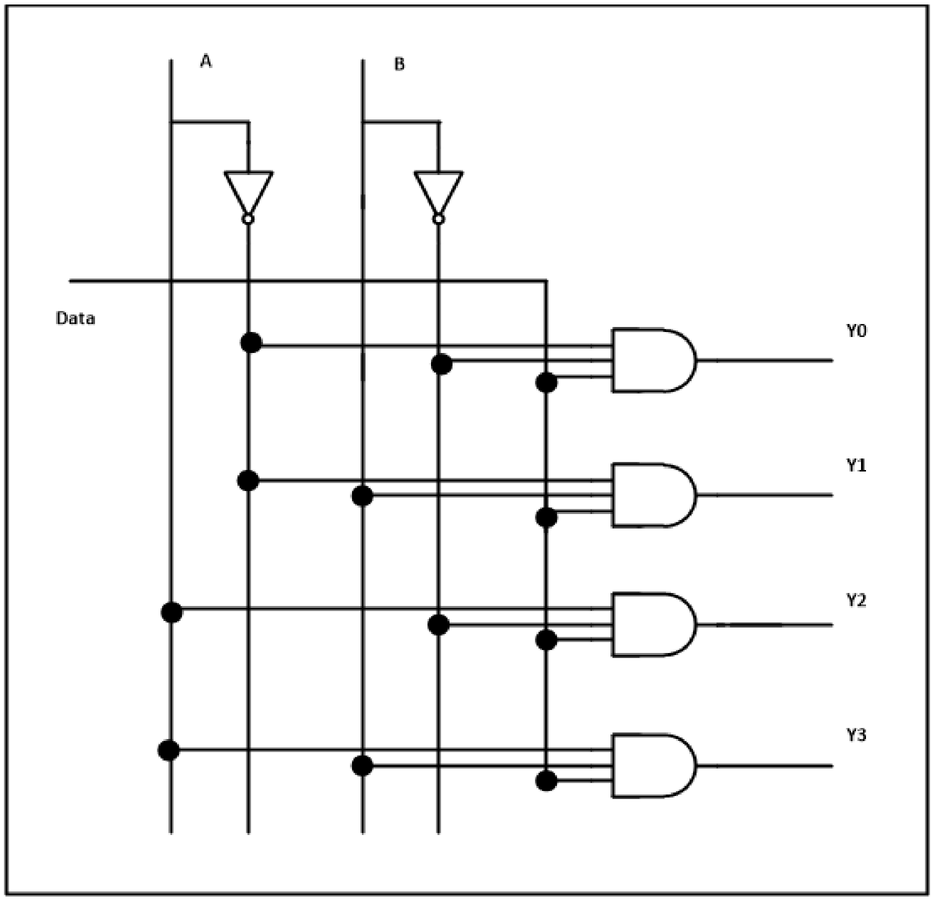
Demultiplexer means one to many. A demultiplexer is a circuit with one input and many output. By applying control signal, we can steer any input to the output. Few types of demultiplexer are 1-to 2, 1-to-4, 1-to-8 and 1-to 16 demultiplexer.

Following figure illustrate the general idea of a demultiplexer with 1 input signal, m control signals, and n output signals.



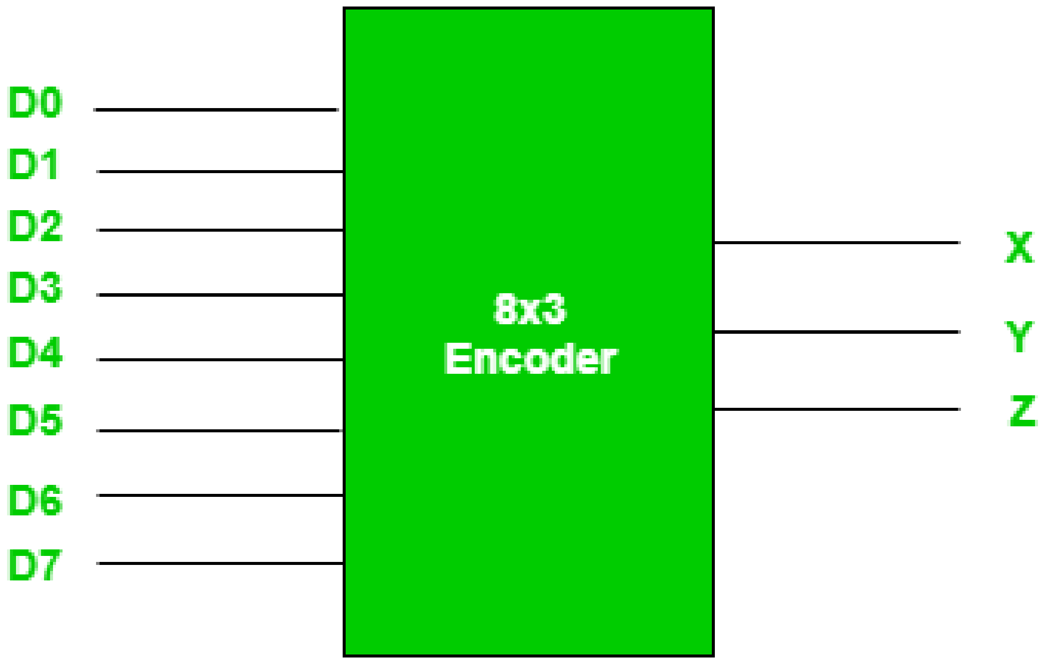
Understanding 1- to-4 Demultiplexer:

The 1-to-4 demultiplexer has 1 input bit, 2 control bit, and 4 output bits. An example of 1-to-4 demultiplexer is IC 74155. The 1-to-4 demultiplexer is shown in figure below-



Encoders and Decoders in Digital Logic:

Binary code of N digits can be used to store 2^N distinct elements of coded information. This is what encoders and decoders are used for. **Encoders** convert 2^N lines of input into a code of N bits and **Decoders** decode the N bits into 2^N lines. **Encoders** –
An encoder is a combinational circuit that converts binary information in the form of a 2^N input lines into N output lines, which represent N bit code for the input. For simple encoders, it is assumed that only one input line is active at a time.
As an example, let’s consider **Octal to Binary** encoder. As shown in the following figure, an octal-to-binary encoder takes 8 input lines and generates 3 output lines.


D0
D1
D2
D3
D4
D5
D6
D7
8x3
Encoder
X
Y
Z

Truth Table –

D7	D6	D5	D4	D3	D2	D1	D0	X	Y	Z
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

As seen from the truth table, the output is 000 when D0 is active; 001 when D1 is active; 010 when D2 is active and so on.

Implementation –
From the truth table, the output line Z is active when the input octal digit is 1, 3, 5 or 7. Similarly, Y is 1 when input octal digit is 2, 3, 6 or 7 and X is 1 for input octal digits 4, 5, 6 or 7. Hence, the Boolean functions would be:

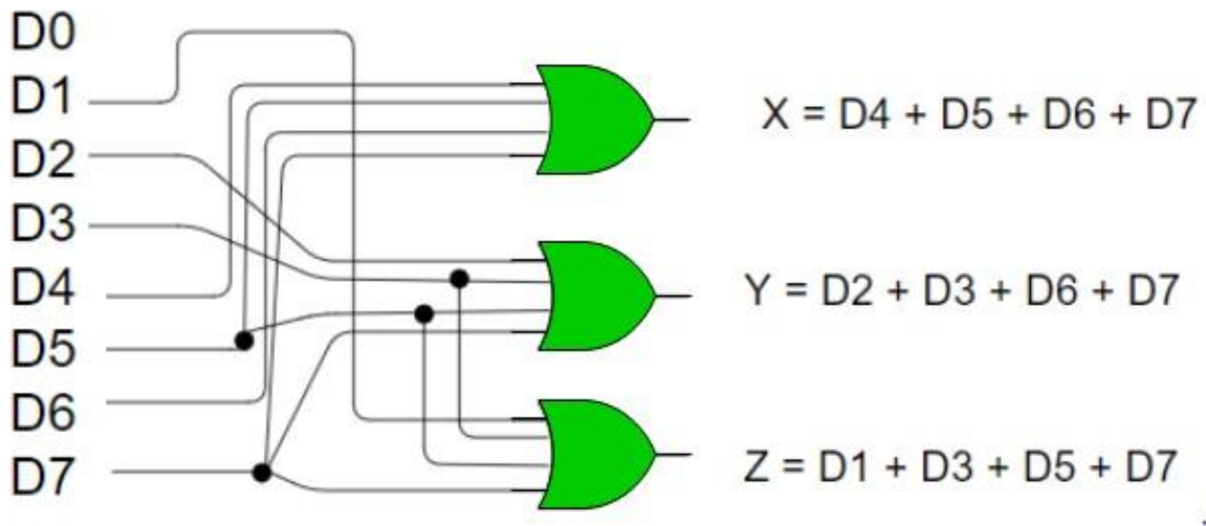
X = D4 + D5 + D6 + D7

Y = D2 +D3 + D6 + D7

Z = D1 + D3 + D5 + D7

Hence, the encoder can be realised with OR gates as follows:

:



One limitation of this encoder is that only one input can be active at any given time. If more than one inputs are active, then the output is undefined. For example, if D6 and D3 are both active, then, our output would be 111 which is the output for D7. To overcome this, we use Priority Encoders.

Another ambiguity arises when all inputs are 0. In this case, encoder outputs 000 which actually is the output for D0 active. In order to avoid this, an extra bit can be added to the output, called the valid bit which is 0 when all inputs are 0 and 1 otherwise.

Decoders :

A decoder does the opposite job of an encoder. It is a combinational circuit that converts n lines of input into 2^n lines of output.
Let’s take an example of 3-to-8 line decoder.

Truth Table –

X	Y	Z	D0	D1	D2	D3	D4	D5	D6	D7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Implementation

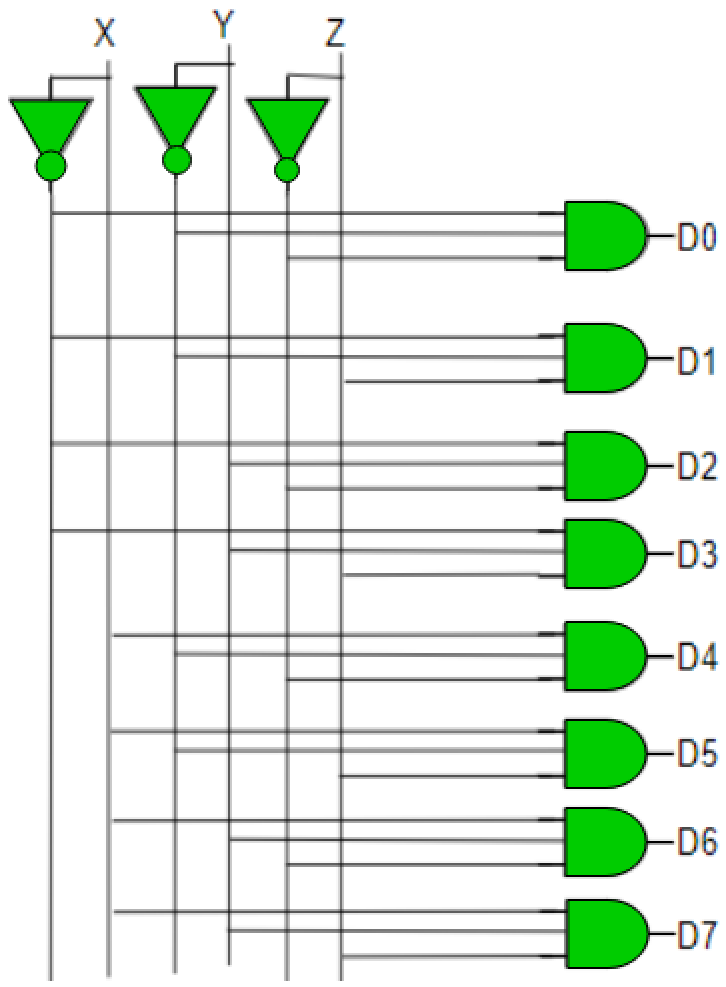
D0 is high when X = 0, Y = 0 and Z = 0. Hence,

```
D0 =  X’  Y’ Z’
```

Similarly,

```
D1  = X’  Y’ Z
D2  = X’  Y Z’
D3  = X’  Y  Z
D4  = X  Y’ Z’
D5  = X  Y’ Z
D6  = X  Y Z’
D7  = X  Y  Z
```

Hence,



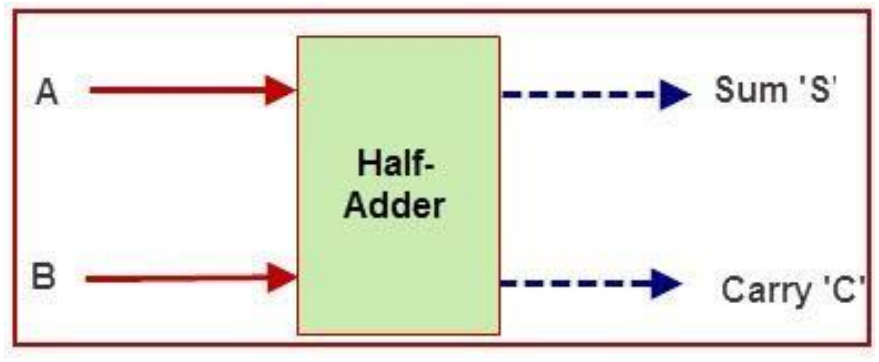
What is Haff Adder and Full Adder?

An **adder** is a device that will add together two bits and give the result as the output. ... A **half adder** just adds two bits together and gives a two-bit output. A **full adder** adds two inputs and a carried input from another **adder**, and also gives a two-bit output.

Half Adder:

By using half adder, you can design simple addition with the help of logic gates.

Let's see an addition of single bits.



Half Adder

0+0 = 0

0+1 = 1

1+0 = 1

1+1 = 10

These are the least possible single-bit combinations. But the result for 1+1 is 10, the sum result must be re-written as a 2-bit output. Thus, the equations can be written as

0+0 = 00

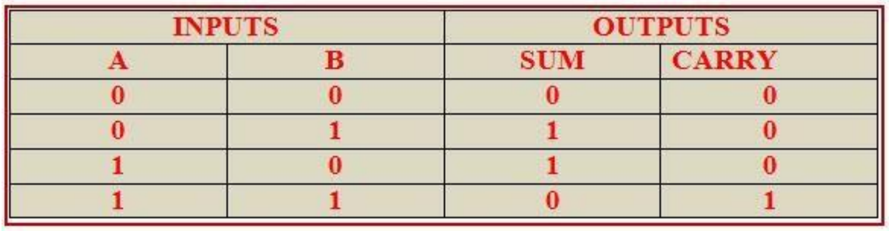
0+1 = 01

1+0 = 01

1+1 = 10

The output ‘1’of ‘10’ is carry-out. ‘SUM’ is the normal output and ‘CARRY’ is the carry-out.

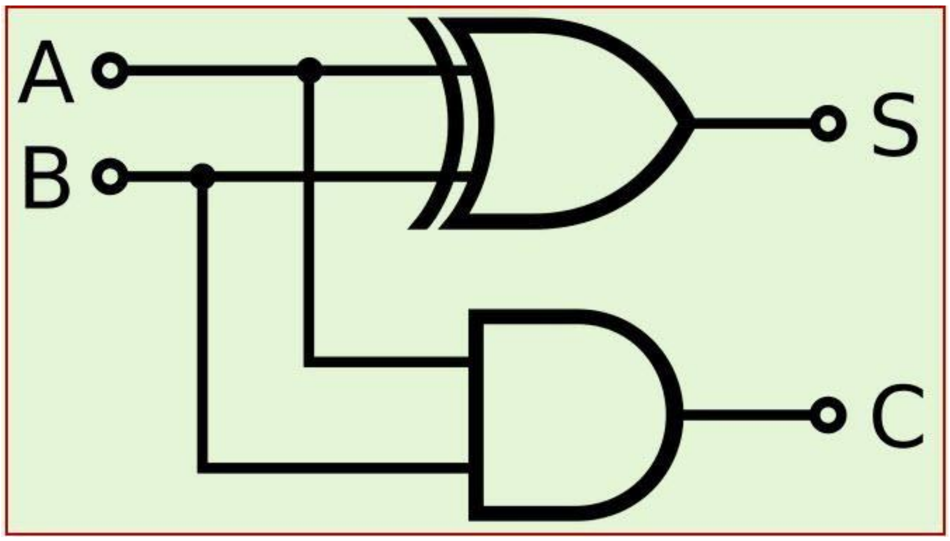
Half Adder Truth Table



INPUTS		OUTPUTS	
A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

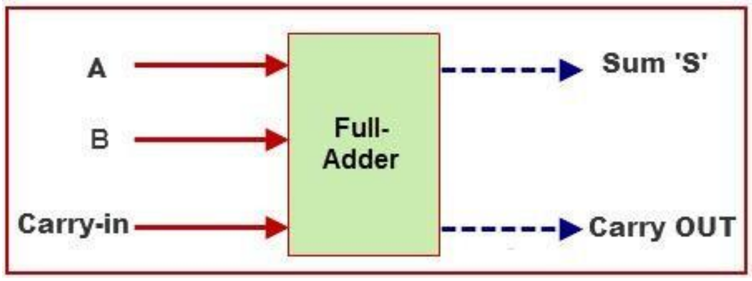
Half Adder Truth Table

Now it has been cleared that 1-bit adder can be easily implemented with the help of the XOR Gate for the output 'SUM' and an AND Gate for the 'Carry'. When we need to add, two 8-bit bytes together, we can be done with the help of a full-adder logic. The half-adder is useful when you want to add one binary digit quantities. A way to develop a two-binary digit adders would be to make a truth table and reduce it. When you want to make a three binary digit adder, do it again. When you decide to make a four digit adder, do it again. The circuits would be fast, but development time is slow.



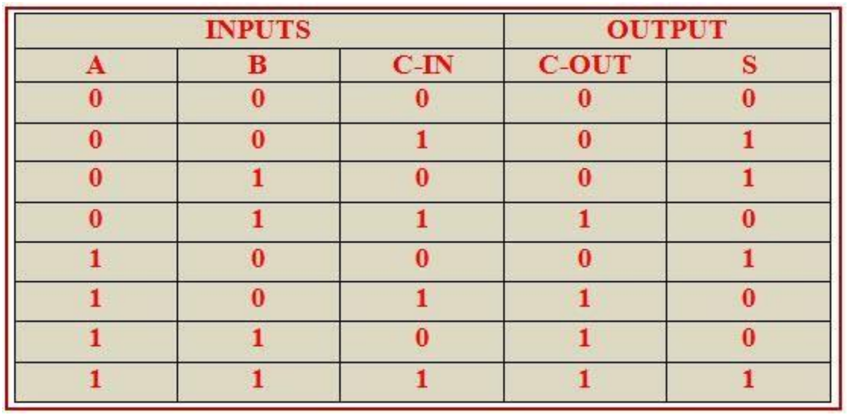
Full Adder:

This adder is difficult to implement than a half-adder. The difference between a half-adder and a full-adder is that the full-adder has three inputs and two outputs, whereas half adder has only two inputs and two outputs. The first two inputs are A and B and the third input is an input carry as C-IN. When a full-adder logic is designed, you string eight of them together to create a byte-wide adder and cascade the carry bit from one adder to the next.



The output carry is designated as C-OUT and the normal output is designated as S.

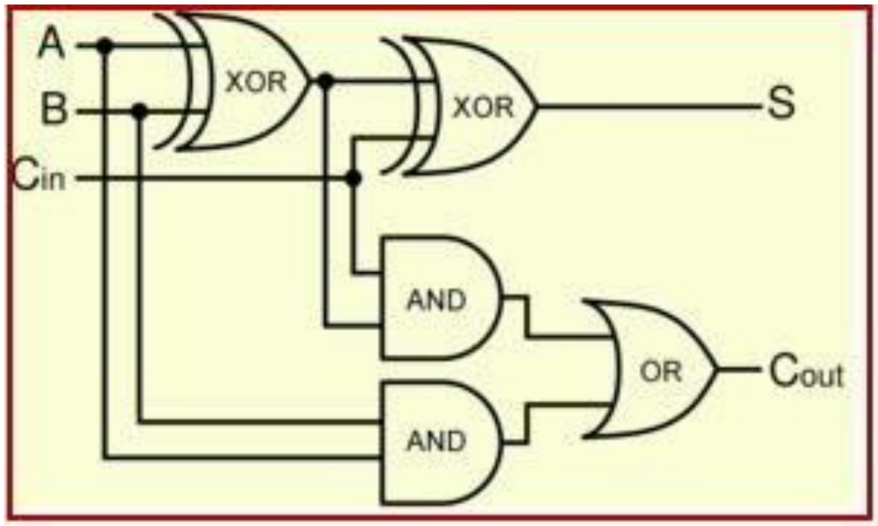
Full Adder Truth Table:



INPUTS			OUTPUT	
A	B	C-IN	C-OUT	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

With the truth-table, the full adder logic can be implemented. You can see that the output S is an XOR between the input A and the half-adder, SUM output with B and C-IN inputs. We take C-OUT will only be true if any of the two inputs out of the three are HIGH.

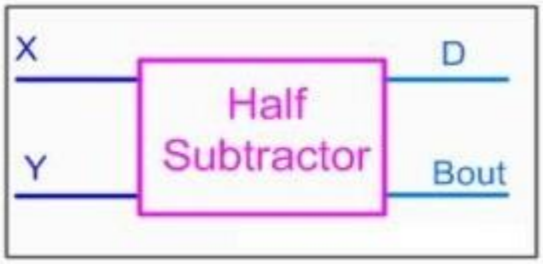
So, we can implement a full adder circuit with the help of two half adder circuits. At first, half adder will be used to add A and B to produce a partial Sum and a second half adder logic can be used to add C-IN to the Sum produced by the first half adder to get the final S output.



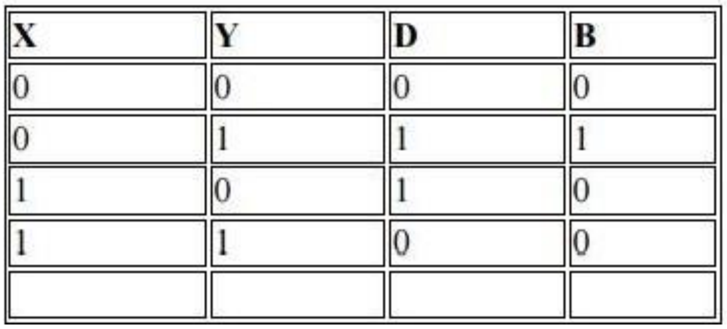
Half Subtractor:

The half-subtractor is a combinational circuit which is used to perform subtraction of two bits. It has two inputs, X (minuend) and Y (subtrahend) and two outputs D (difference) and B (borrow). The logic symbol and truth table are shown below.

Symbol

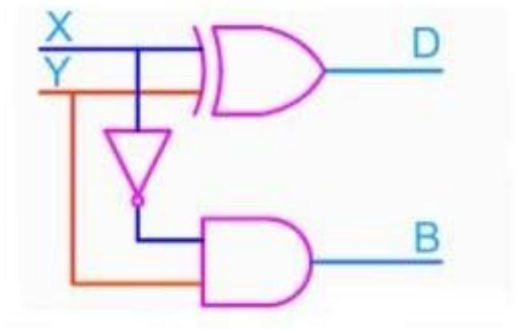


Truth Table



X	Y	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

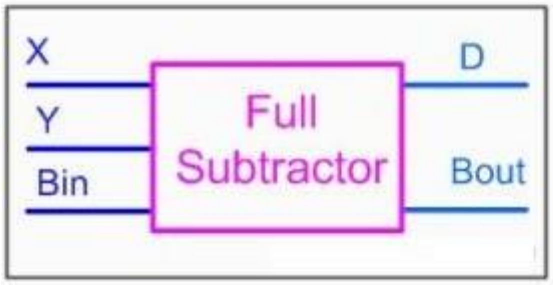
we can draw the half-subtractor as shown in the figure below.



Full Subtractor

A full subtractor is a combinational circuit that performs subtraction involving three bits, namely minuend, subtrahend, and borrow-in. The logic symbol and truth table are shown below.

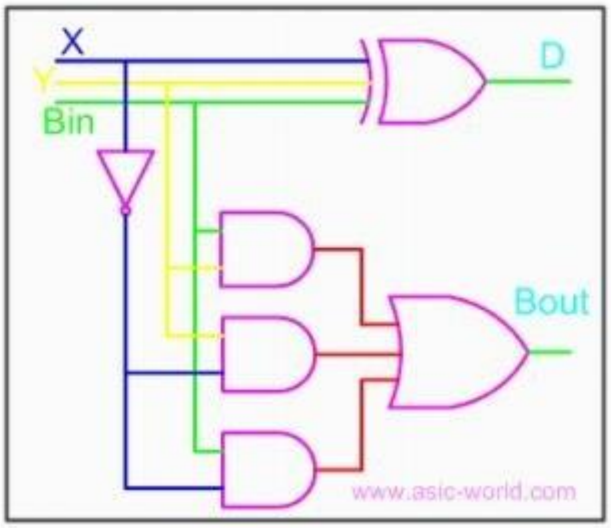
Symbol



Truth Table

X	Y	Bin	D	Bout
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Full-subtractor circuit is more or less same as a full-adder with slight modification.



UNIT IV

Sequential Logic Circuits: Flip Flops – RS Flip Flops – D Flip Flops- T Flip Flops – JK Flip-flops - * Shift Registers (Serial-In-Serial-Out)*. *……* Self Study

What is Flip-flops?

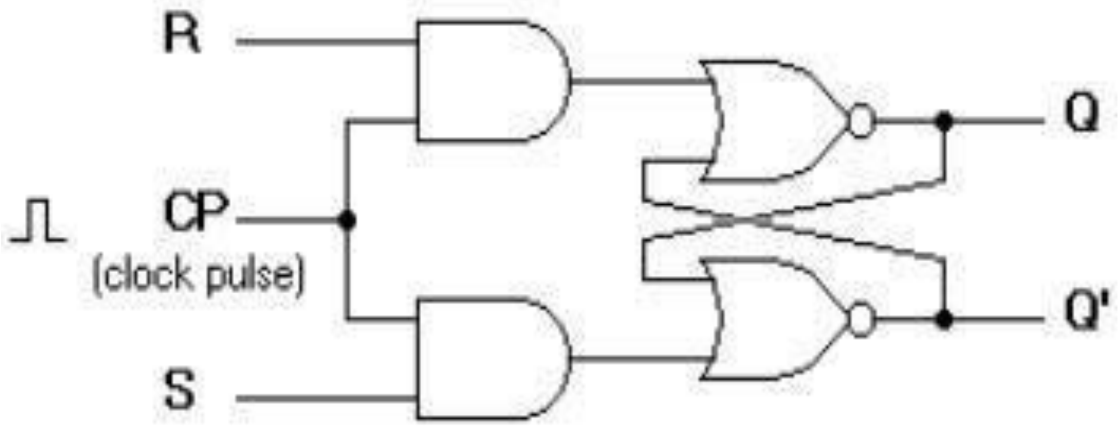
In electronics, a **flip-flop** is a special **type** of gated latch circuit. There are several different **types** of **flip-flops**. The most common **types** of **flip flops** are: ... Besides the CLOCK input, an SR **flip-flop** has two inputs, labeled SET and RESET. If the SET input is HIGH when the clock is triggered, the Q output goes HIGH.

S-R Flip Flop:

The SET-RESET flip flop is designed with the help of two NOR gates and also two NAND gates. These flip flops are also called S-R Latch.

S-R Flip Flop using NOR Gate

The design of such a flip flop includes two inputs, called the SET [S] and RESET [R]. There are also two outputs, Q and Q’. The diagram and truth table is shown below.



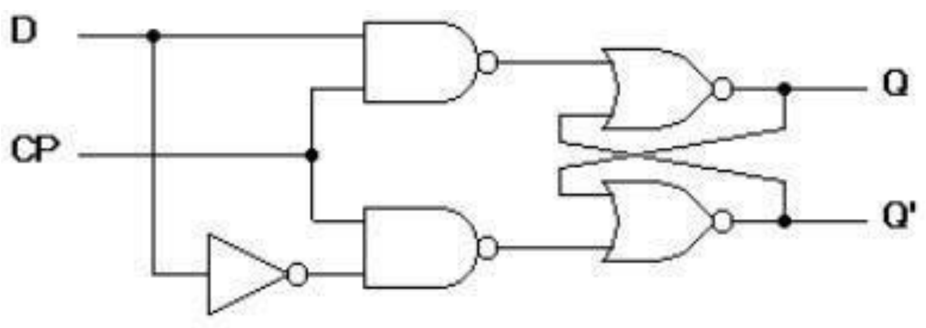
(a) Logic diagram

Q	S	R	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	indeterminate
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	indeterminate

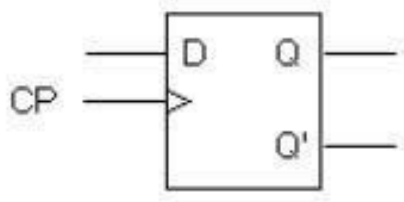
(b) Truth table

Clocked SR flip-flop

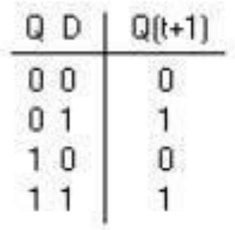
D Flip Flop:
The circuit diagram and truth table is given below.



(a) Logic diagram with NAND gates



(b) Graphical symbol



Q D	Q(t+1)
0 0	0
0 1	1
1 0	0
1 1	1

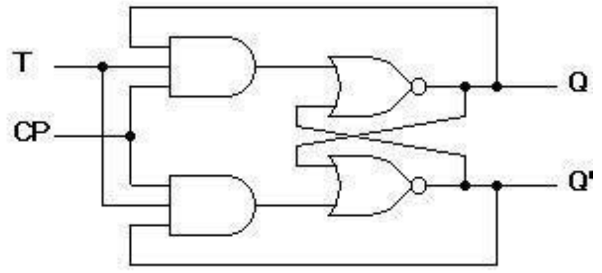
(c) Transition table

Clocked D flip-flop

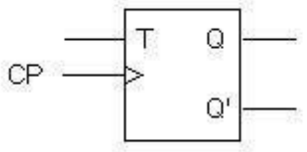
D flip flop is actually a slight modification of the above explained clocked SR flip-flop. From the figure you can see that the D input is connected to the S input and the complement of the D input is connected to the R input. The D input is passed on to the flip flop when the value of CP is ‘1’. When CP is HIGH, the flip flop moves to the SET state. If it is ‘0’, the flip flop switches to the CLEAR state.

T Flip Flop:

This is a much simpler version of the J-K flip flop. Both the J and K inputs are connected together and thus are also called a single input J-K flip flop. When clock pulse is given to the flip flop, the output begins to toggle. Here also the restriction on the pulse width can be eliminated with a master-slave or edge-triggered construction. Take a look at the circuit and truth table below.



(a) Logic diagram



(b) Graphical symbol

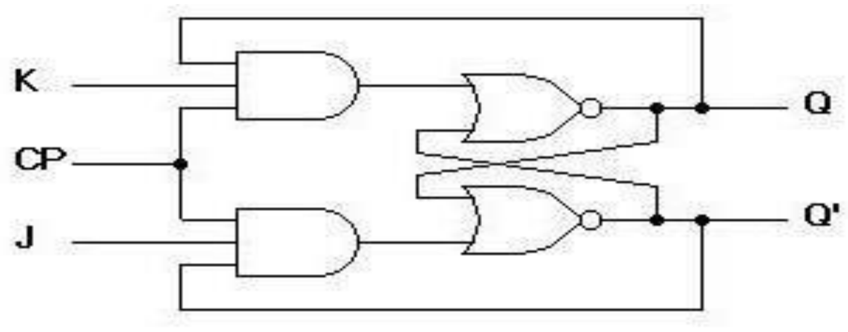
Q T	Q(t+1)
0 0	0
0 1	1
1 0	1
1 1	0

(c) Transition table

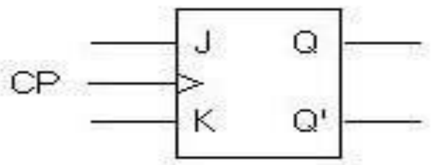
Clocked T flip-flop

J-K Flip Flop:

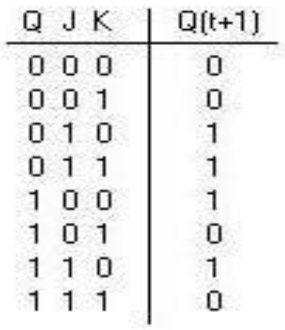
The circuit diagram and truth-table of a J-K flip flop is shown below.



(a) Logic diagram



(b) Graphical symbol



Q	J	K	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

(c) Transition table

Clocked JK flip-flop

J-K Flip Flop

A J-K flip flop can also be defined as a modification of the S-R flip flop. The only difference is that the intermediate state is more refined and precise than that of a S-R flip flop.

The behavior of inputs J and K is same as the S and R inputs of the S-R flip flop. The letter J stands for SET and the letter K stands for CLEAR.

When both the inputs J and K have a HIGH state, the flip-flop switch to the complement state. So, for a value of Q = 1, it switches to Q=0 and for a value of Q = 0, it switches to Q=1.

The circuit includes two 3-input AND gates. The output Q of the flip flop is returned back as a feedback to the input of the AND along with other inputs like K and clock pulse [CP]. So, if the value of CP is '1', the flip flop gets a CLEAR signal and with the condition that the value of Q was earlier 1. Similarly output Q' of the flip flop is given as a feedback to the input of the AND along with other inputs like J and clock pulse [CP]. So the output becomes SET when the value of CP is 1 only if the value of Q' was earlier 1.

The output may be repeated in transitions once they have been complimented for J=K=1 because of the feedback connection in the JK flip-flop. This can be avoided by setting a time duration lesser than the propagation delay through the flip-flop. The restriction on the pulse width can be eliminated with a master-slave or edge-triggered construction.

What is Serial in and Serial out shift registers?

Serial In Serial Out (SISO) **shift** registers are a kind of **shift** registers where both data loading as well as data retrieval to/from the **shift register** occurs in **serial**-mode .... Here the data word which is to be stored is fed bit-by-bit at the input of the first flip-flop.

Serial In Serial Out (SISO) shift registers are a kind of shift registers where both data loading as well as data retrieval to/from the shift register occurs in serial-mode. Figure 1 shows a n-bit synchronous **SISO shift register** sensitive to positive edge of the clock pulse. Here the data word which is to be stored is fed bit-by-bit at the input of the first flip-flop. Further it is seen that the inputs of all other flip-flops (except the first flip-flop FF_1) are driven by the outputs of the preceding ones say for example, the input of FF_2 is driven by the output of FF_1. At last the data stored within the register is obtained at the output pin of the nth flip-flop in serial-fashion.

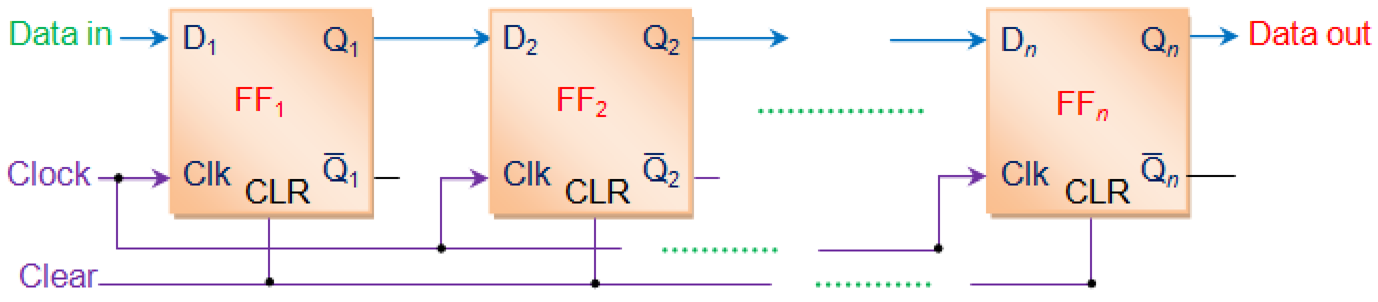


Figure 1 *n*-bit Right-Shift Serial-in Serial-Out Shift Register

Initially all the flip-flops in the register are cleared by applying high on their clear pins. Next the input data word is fed serially to FF_1.
This causes the bit appearing at the D_1 pin (B_1) to be stored into FF_1 as soon as the first leading edge of the clock appears. Further at the second clock tick, B_1 gets stored into FF_2 while a new bit enters into FF_1 (B_2).

This kind of shift in data bits continues for every rising edge of the clock pulse. This indicates that for every single clock pulse the data within the register moves towards right by a single bit. Thus the design shown in Figure 1 is regarded as a right-shift **SISO shift register**. Following the data transmission as explained, one can note that the first bit of an input word appears at the output of nth flip-flop for the nth clock tick. On applying further clock cycles, one gets the next successive bits of the input data word as the serial output.

(Table I). The waveforms pertaining to the same are shown by Figure 2.

Clock Cycle	Data in	Q_1	Q_2	…	Q_n = Data out
1	B_1	B_1	0	…	0
2	B_2	B_2	B_1	…	0
3	B_3	B_3	B_2	…	0
4	B_4	B_4	B_3	…	0
5	B_5	B_5	B_4	…	0
6	B_6	B_6	B_5	…	0
⋮	⋮	⋮	⋮	…	⋮
n	B_n	B_n	B_{n-1}	…	B_1
$n+1$	B_{n+1}	B_{n+1}	B_n	…	B_2
⋮	⋮	⋮	⋮	…	⋮

Serial Output Bits of SISO (Right-Shift) Shift Register

Table I **Data Movement in Right-Shift SISO Shift**

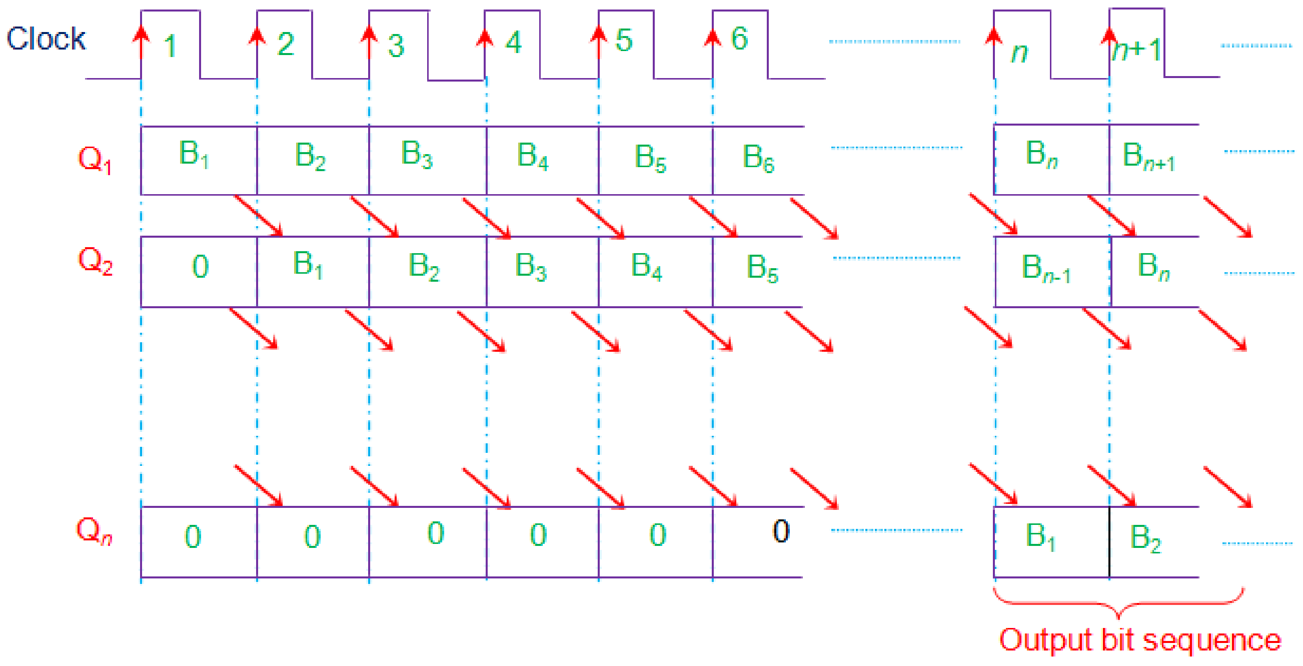


Figure 2 **Output Waveform of *n*-bit Right-Shift SISO Shift Register**

UNIT – V

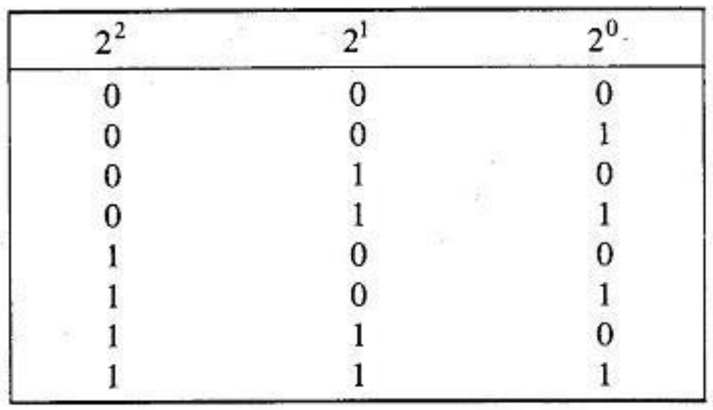
D/A and A/D Conversion – Variable Resistor Network – Binary Ladder – D/A Converter – D/A Accuracy and Resolution – A/D Converters – *Simultaneous Method *.

Variable Resistor Network:

Variable Resistor Network – The basic problem in converting a digital signal into an equivalent analog signal is to change the n digital voltage levels into one equivalent analog voltage. This can be achieved most easily by designing a Variable Resistor Network which changes each of the digital levels into an equivalent binary weight voltage (or current). To understand the meaning of equivalent binary weight consider the truth table for the 3 bit binary signal shown in Table 17.1.

Suppose we wish to change the 8 possible states of digital signals into equivalent analog voltages. The smallest number represented by 000 is OV and the largest number represented is 111. Let us make this signal equal to + 7V. This then establishes the range of the analog signal which will be developed. Now, between 000 and 111 there are seven discrete levels to be defined. Therefore, it is convenient to divide the analog signal into seven levels.

Table 17.1 Truth Table for a 3-bit Binary



2^2	2^1	2^0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

.

The smallest increment change in the digital signal is represented by the LSB (2^0). Hence we would like to have this bit cause a change in the analog output equal to 1/7 of the full scale analog output voltage. The resistive divider will then be designed such that a 1 in the 2^0 position causes + 7 x 1/7 = 1 V at the output.

Since $2^1 = 2$ and $2^0 = 1$, it can be seen that the 2^1 number is twice the size of the 2^0 bit. Therefore a 1 in the 2^1 bit position must cause a change in the analog output voltage which is twice the size of the LSB. The resistive divider must then be designed such that a 1 in the 2^1 bit position causes a change of + 7 x 2/7 = + 2 V in the analog voltage.

Similarly, $2^2 = 4 = 2^1 \times 2 = 4 \times 2^0$ this 2^2 bit must cause a change in the output voltage which is 4 times that of the LSB. The 2^2 bit must then cause an output voltage change of + 7 x 4/7 = + 4 V.

The process can be continued and it will be seen that each successive bit must have a value which is twice that of the preceding bit. Thus the LSB is given a binary equivalent weight of 1/7, or 1 part in 7. The next LSB is given a weight of 2/7, which is twice the LSB or 2 parts in 7. The MSB (in the case of a 3 bit system) is given by 4/7, which is 4 times the LSB or 4 parts in 7.

The total sum of the weights must be equal to 1. Hence

$1/7+2/7+4/7=7/7=1$

In general, the binary equivalent weights assigned to the LSB is $1/2^n - 1$ where n is the number of bits.

A resistive divider having three digital inputs and an analog output is shown

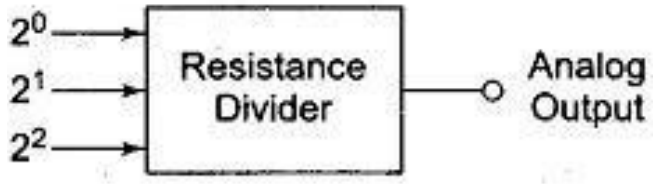


Fig. 17.9

In this Variable Resistor Network case, assume digital levels 0 = 0 V and 1 = + 7 V.

For an input of 001, the output is + 1 V; similarly for 010, the output is + 2 V and for 100, the output is + 4 V. Now digital input of 011 is seen to be a combination of signals 001 and 010. If + 1 V from the 2^0 bit is added to + 2 V from the 2^1 bit, the output is + 3 V for an input of 011. Similarly, other levels are determined by an additive combination of voltages, as shown

Table 17.2 Analog Outputs Levels for a 3-bit Digital Input

Digital inputs	*Analog outputs*
000	0
001	+1V
010	+2V
011	+3V
100	+4V
101	+5V
110	+6V
111	+7V

Binary Ladder Circuits:

A Binary Ladder Circuits is constructed of resistors having only two values and thus overcomes the disadvantages of weighted resistors. The left end of the ladder is terminated in 2 R**,** as shown in Fig. 17.14.

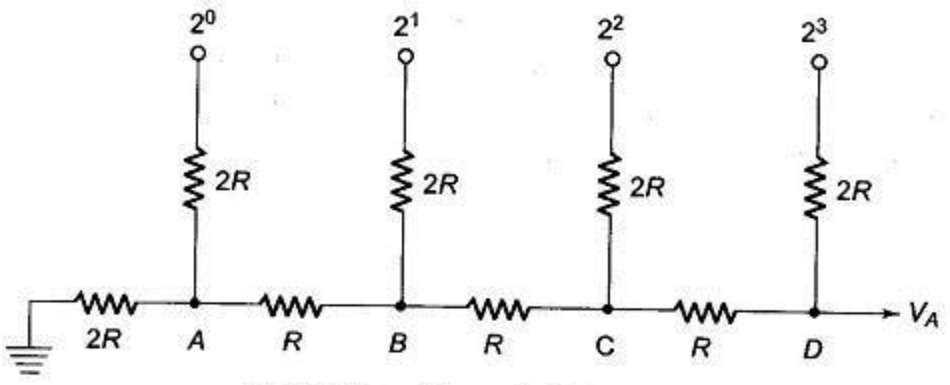


Fig. 17.14 Binary Ladder

Assuming that all digital inputs are at ground, beginning at point A of Fig. 17.14, the total resistance looking into the terminating resistors is 2R, as seen from Fig. 17.15(a). The total resistance looking out towards the 2^0 input is 2R. These two resistors combine to form a value of R**.** At node B or at 2^1, the input is still 2R, as seen from Fig. 17.15(b). It is clear from Fig. 17.15(c) that the resistance looking back towards node C is 2R, as is the resistance looking at the 2^3 input.

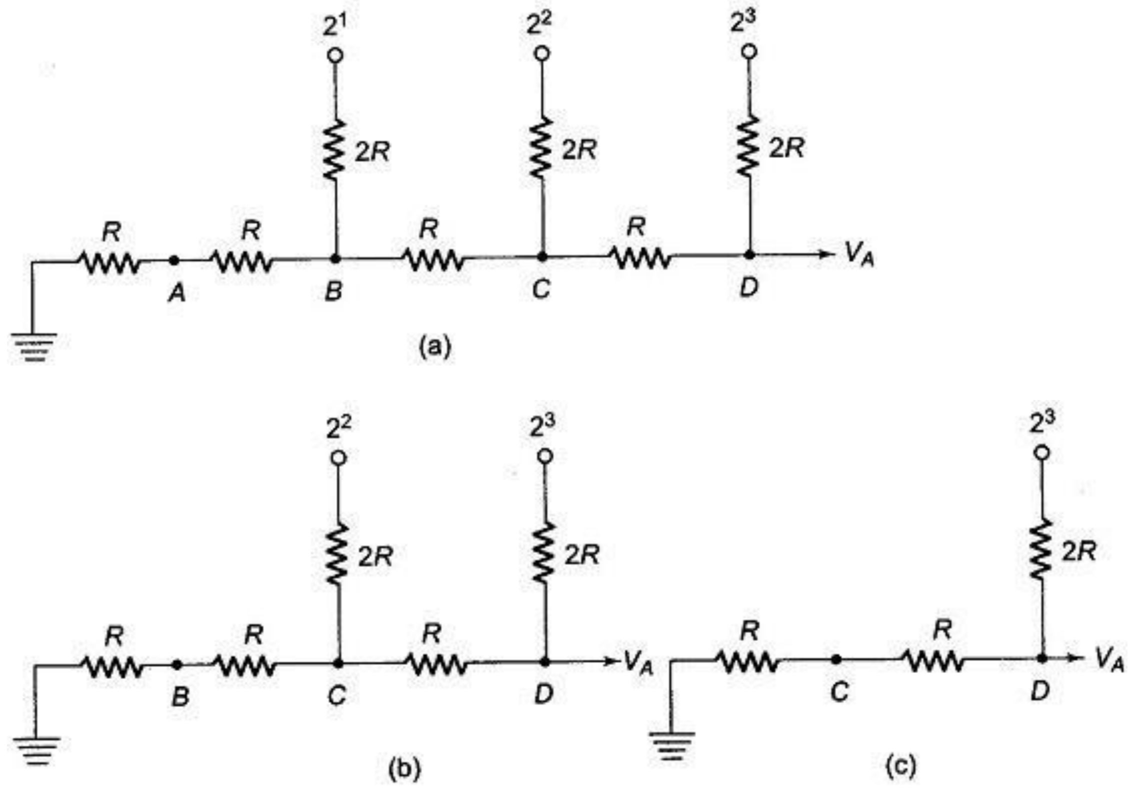


Fig. 17.15 (a) Resistance Looking at Point *A* with Respect to Ground
(b) Resistance Looking at Point *B* with Respect to Ground
(c) Resistance Looking at Point *C* with Respect to Ground

From this we conclude that the resistance looking back from any node towards the terminating resistance or out toward the digital input is 2R. We can use this to determine the various digital inputs. First, assume that the digital input signal is 1000, as shown in Fig. 17.16.

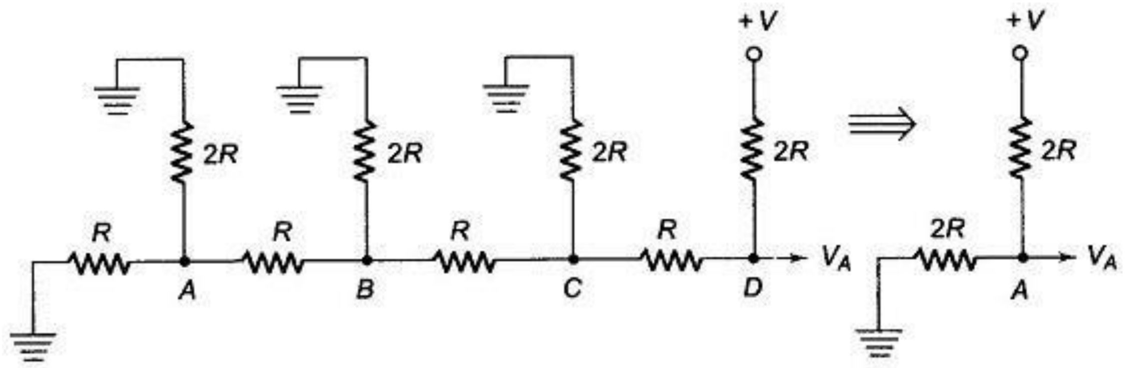


Fig. 17.16 Equivalent Circuit for Binary Number 1000

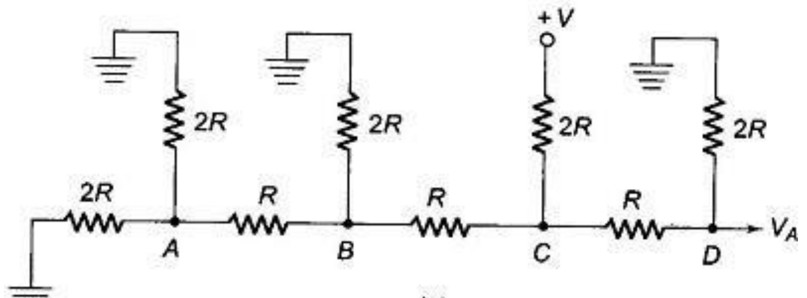
Since there is no voltage source to the left of the node D, it can be replaced by 2R.

Therefore

$$V_A = \frac{2R}{2R+2R} \times V = \frac{+V}{2}$$

Hence a 1 in the MSB position provides an output voltage of +V/2.

To determine the output voltage due to the 2nd MSB, assume an input of 0100, as shown in Fig. 17.17(a).



(a)

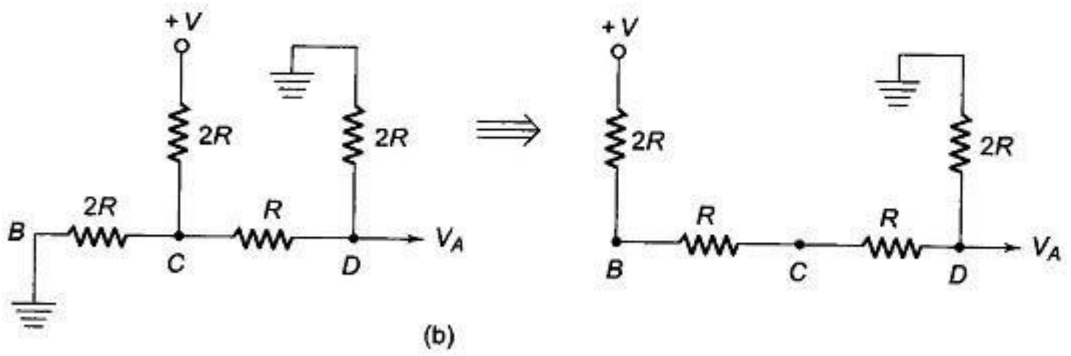


Fig. 17.17 (a) R–2R Ladder Network for Binary 0100
(b) Equivalent Circuit for Binary 0100

Therefore, the left side network of node C with its The'venin equivalent, is clearly a resistance R in series with a voltage source of +V/2. The final equivalent circuit, the The'venin's equivalent include, is

$$V_A = \frac{2R}{2R+2R} \times \frac{+V}{2} = \frac{+V}{4}$$

Hence the 2nd MSB equals + V/4. This process can be continued, and it can be shown that the 3rd MSB gives an output voltage of + V/8, the 4th MSB gives an output voltage of +V/16 and so on.

The various output voltages for the corresponding MSB are given in Table 17.4.

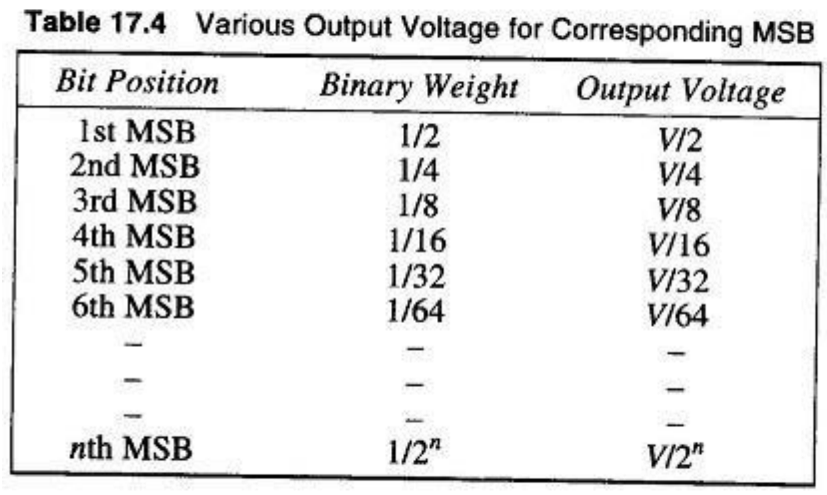


Table 17.4 Various Output Voltage for Corresponding MSB

Bit Position	*Binary Weight*	*Output Voltage*
1st MSB	1/2	V/2
2nd MSB	1/4	V/4
3rd MSB	1/8	V/8
4th MSB	1/16	V/16
5th MSB	1/32	V/32
6th MSB	1/64	V/64
–	–	–
–	–	–
–	–	–
*n*th MSB	$1/2^n$	$V/2^n$

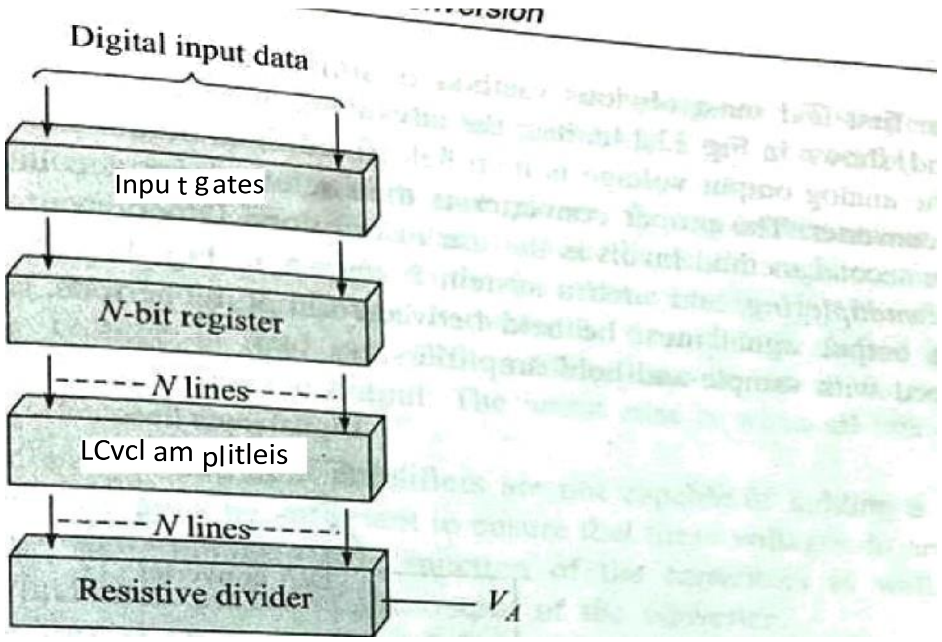
D/A CONVERTERS:

The resistive divider or the ladder can be used as the basis for a digital-to-analog converter. It is in the resistive network that the actual translation from a digital signal to an analog voltage takes place.

As an integral part of the D/A converter there must be a register that can be used to store the digital information. The simplest register is formed by use of RS flip flops, with one flip flop per bit. There must also be level amplifiers between the register and the resistive network to ensure that the digital signals presented to the network are all of the same level and are constant.

The level amplifiers each have two inputs ,one is the +10V from the precision voltage source, and the other is from a flip flop. The amplifiers work in such a way that when the input from a flip flop is high, the output of the amplifier is at +10V. When the input from the flip flop is low, the output is 0V.

The four flip flop form the register necessary for storing the digital information. The flip flop on the right represents the MSB and the flip flop on the left represents the LSB. Each flip flop is simple RS latch and requires a positive level at the R or S input te reset or set it.



(a)

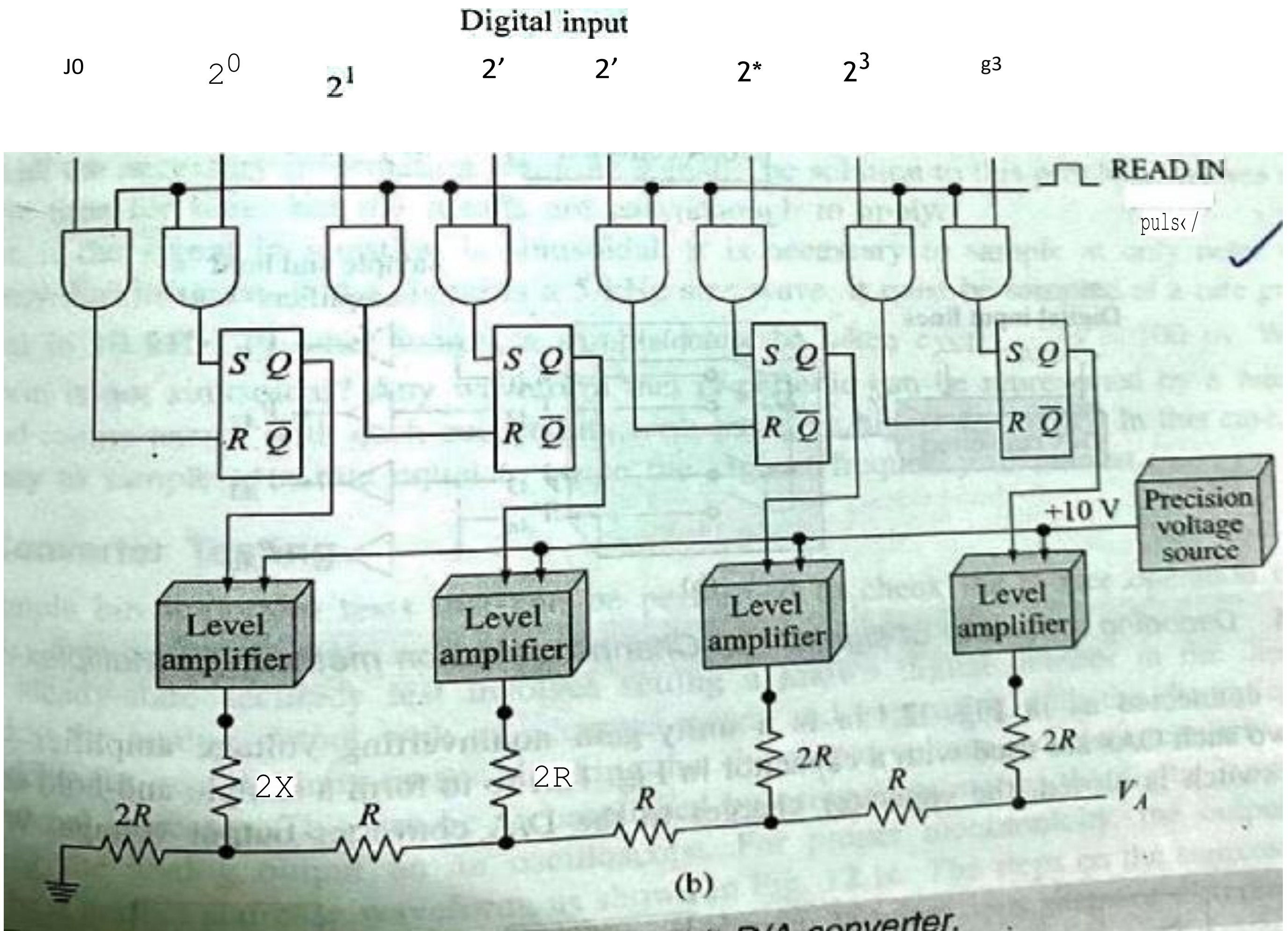


Fig. 12.13 4-bit D/A converter.

D/A ACCURACY AND RESOLUTION:

Two very important aspects of the D/A converter are the resolution and the accuracy of the conversion.

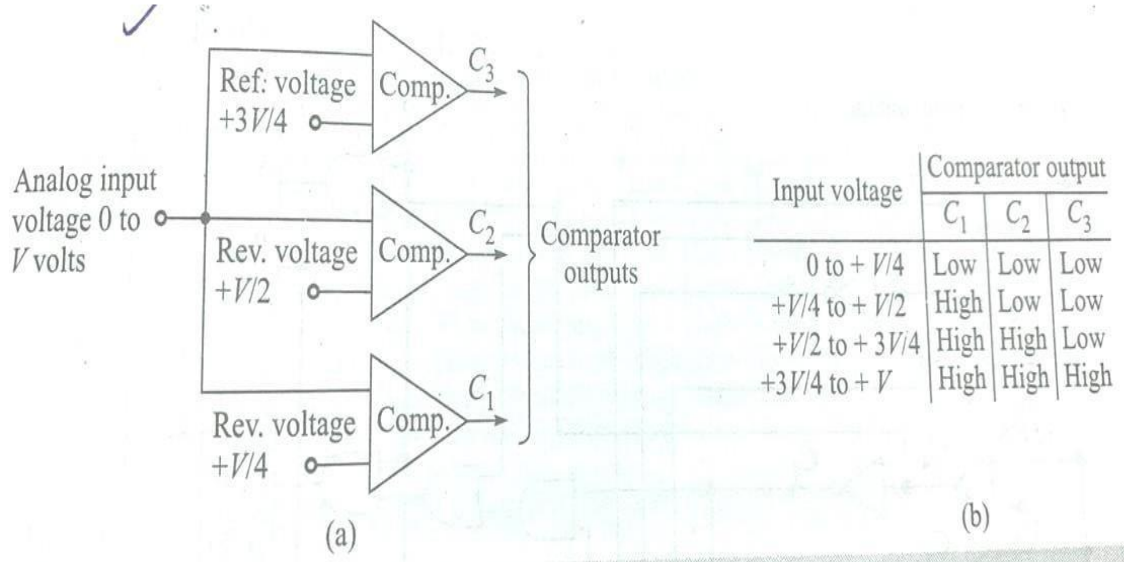
The accuracy of the D/A converter is primarily a function of the accuracy of the precision resistors used in the ladder and the precision of the reference voltage supply used. Accuracy is measure of how close the actual output voltage is to the theoretical output value.

Resolution on the other hand, defines the smallest increment in voltage that can be discerned. Resolution is primarily a function of the number of bits in the digital input signal that is the smallest increment in output voltage is determined by the LSB.

If we wanted to represent voltages to a finer resolution we would have to use a converter with more input bits As an example, the LSB of a 10-bit converter has a weight of 1/1024. Thus the smallest incremental change in the output of this converter is approximately 1/1000 of the full scale voltage.

A/D CONVERTER-SIMULTANEOUS CONVERSION:

The process of converting an analog voltage into an equivalent digital signal is known as analog to digital conversion. The simultaneous method of A/D conversion is based on the use of a number of comparator circuits.



Input voltage	Comparator output		
	C_1	C_2	C_3
0 to + V/4	Low	Low	Low
+V/4 to + V/2	High	Low	Low
+V/2 to + 3V/4	High	High	Low
+3V/4 to + V	High	High	High

Fig. 12.19 *Simultaneous A/D conversion. (a) Logic diagram. (b) Comparator outputs for input voltage ranges.*

The analog signal to be digitized serves as one of the inputs to each comparator. The second input is a standard reference voltage. The reference voltages used are +V/4, +V/2 and +3V/4. The system is then capable of accepting an analog input voltage between 0 and +V.

If the analog input signal exceeds the reference voltage to any comparator, that comparator turns on, if all the comparators are off, the analog input signal must be between 0 and +V/4.

If C1 is high and C2 and C3 are low, the input must be between +V/4 and +V/2 V. If C1 and C2 are high while C3 is low, the input must be between +3V/4 and +V