SAMPLE OF THE STUDY MATERIAL
PART OF CHAPTER 5
Combinational & Sequential Circuits

5.1 Introduction

Digital circuits can be classified into two types:

→ Combinational digital circuits and

→ Sequential digital circuits.

5.2 Combinational Digital Circuits

- **Combination Logic Circuits** are made up from basic gates (AND, OR, NOT) or universal gates (NAND, NOR) gates that are "combined" or connected together to produce more complicated switching circuits. These logic gates are the building blocks of combinational logic circuits. An example of a combinational circuit is a decoder, which converts the binary code data present at its input into a number of different output lines, one at a time producing an equivalent decimal code at its output.
- In these circuits "the outputs at any instant of time depends on the inputs present at that instant only."
- For the design of Combinational digital circuits Basic gates (AND, OR, NOT) or universal gates (NAND, NOR) are used. Examples for combinational digital circuits are Half adder, Full adder, Half subtractor, Full subtractor, Code converter, Decoder, Multiplexer, Demultiplexer, Encoder, ROM, etc.

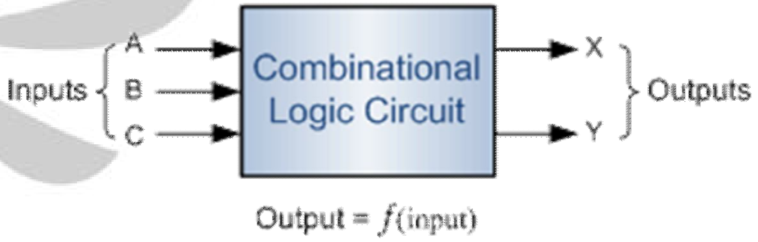


Fig. 5.1 Combinational Digital Circuit

5.3 Classification of Combinational Logic

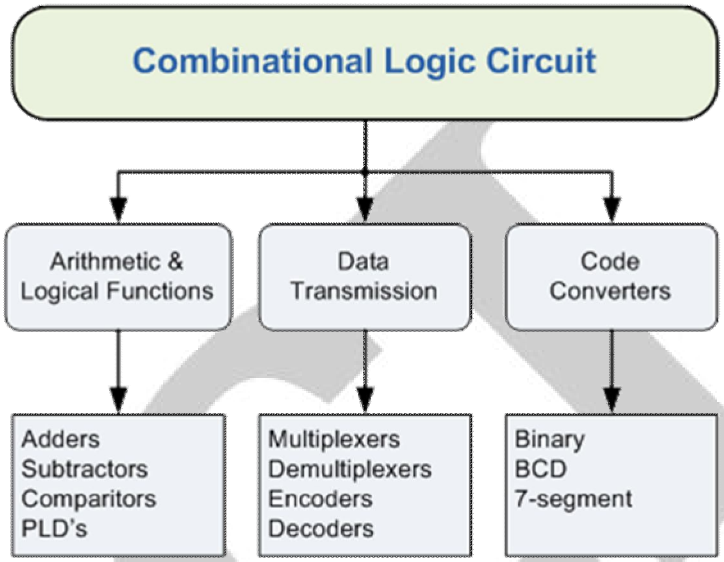


Fig.5.2 Combinational Digital Circuit

5.4 Sequential Digital Circuits

- Sequential logic differs from combinational logic in that the output of the logic device is dependent not only on the present inputs to the device, but also on past inputs; *i.e.*, the output of a sequential logic device depends on its present internal state and the present inputs. This implies that a sequential logic device has some kind of *memory* of at least part of its ``history" (*i.e.*, its previous inputs).
- A simple memory device can be constructed from combinational devices with which we are already familiar. By a memory device, we mean a device which can remember if a signal of logic level 0 or 1 has been connected to one of its inputs, and can make this fact available at an output. A very simple, but still useful, memory device can be constructed from a simple OR gate, as shown in Figure below:

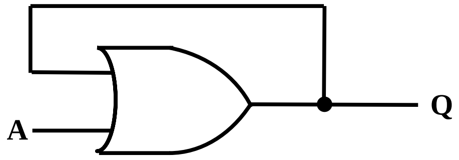


Fig. 5.3 Sequential Digital Circuit

In this memory device, if A and Q are initially at logic 0, then Q remains at logic 0. However if the single input A ever becomes a logic 1, then the output Q will be logic 1 ever after, regardless of any further changes in the input at A. In this simple memory, the output is a function of the state of the memory element only; after the memory is ``written'' then it cannot be changed back. However, it can be ``read.'' Such a device could be used as a simple read only memory, which could be ``programmed'' only once. Often a *state table* or *timing diagram* is used to describe the behavior of a sequential device. Figure 5.4 shows both a state table and a timing diagram for this simple memory. The state table shows the state which the device enters after an input (the ``next state''), for all possible states and inputs. For this device, the output is the value stored in the memory

State table			
Present State	**Input A**	**Next State**	**Output**
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	1

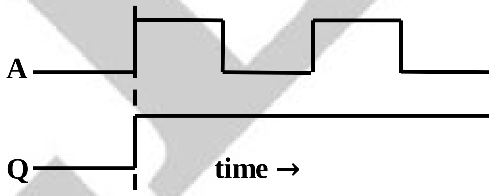


Fig. 5.4 Sequential Circuit Timing Diagram

- Note that the output of the memory is used as one of the inputs; this is called *feedback* and is characteristic of programmable memory devices. (Without feedback, a ``permanent'' electronic memory device would not be possible.) The use of feedback in a device can introduce problems which are not found in strictly combinational circuits. In particular, it is possible to inadvertently construct devices for which the output is not determined by the inputs, and for which it is not possible to predict the output. A simple example is an inverter with its input connected to its output. Such a device is logically inconsistent; in a physical implementation the device would probably either oscillate from 1 to 0 to 1 ⋯ or remain at an intermediate value between logic 0 and logic 1, producing an invalid and erroneous output.
- Examples for sequential digital circuits are Registers, Shift register, Counters etc.

→ **Half Adder:** A half adder is a logical circuit that performs an addition operation on two binary digits. The half adder produces a sum and a carry value which are both binary digits.

X	Y	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Sum = $X \oplus Y = XY' + X'Y$

Carry = XY

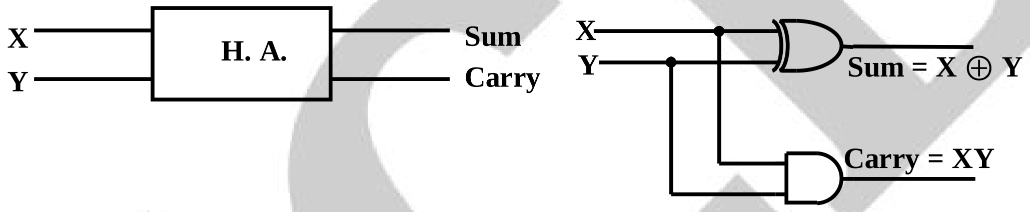


Fig 5.5 Half Adder

→ **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 Difference and Borrow.

X	Y	Borrow	Diff.
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

Diff. $= X \oplus Y = XY' + X'Y$

Borrow $= X'Y$

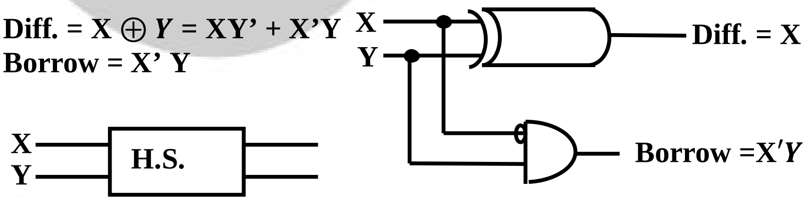


Fig. 5.6 Half Subtractor

→ Half adder can be converted into half subtractor with an additional inverter.

→ **Full adder:** Full adder circuit adds three bit binary numbers (X,Y,Z) & outputs two nos. of one bit binary numbers, Sum & Carry.

X	Y	Z	Carry	Sum
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

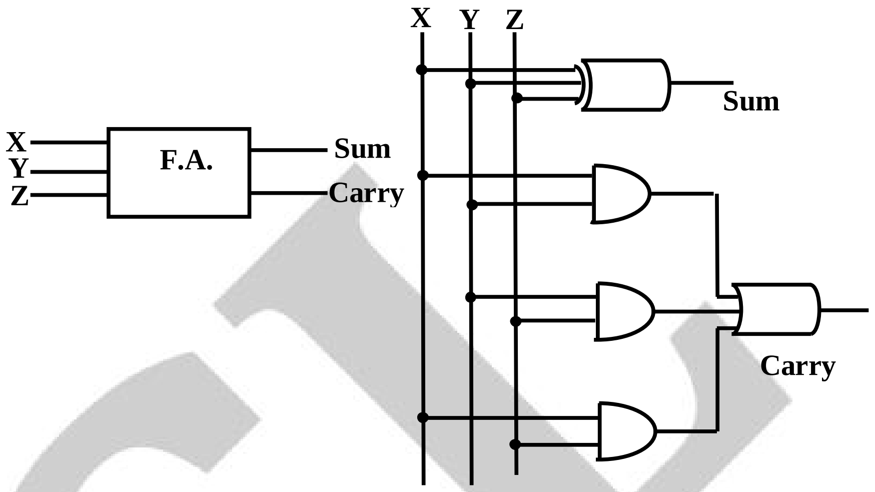
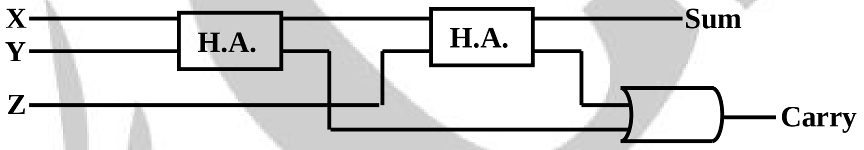


Fig. 5.7 Full Adder

→ Full adder can be implemented by using two half adders and an OR gate.



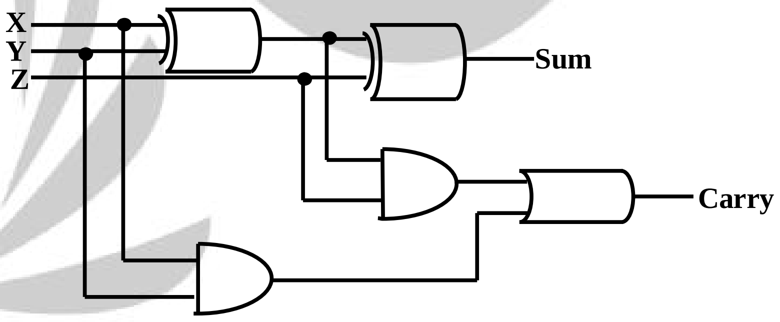
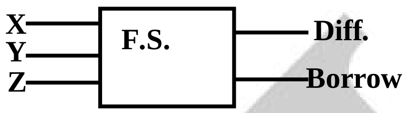


Fig. 5.8 Full Adder

→ **Full subtractor:** It subtracts one bit from the other by taking pervious borrow into account and generates difference and borrow

Truth Table X – Y – Z

X	Y	Z	Borrow	Diff.
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1



Diff. $= X \oplus Y \oplus Z$
Borrow = X'Y + YZ + X'Z

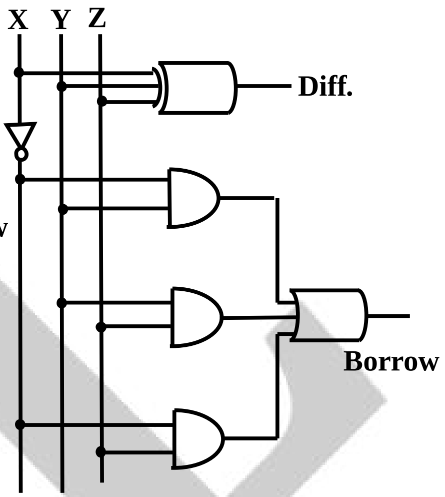
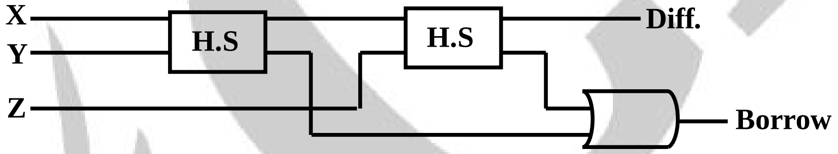


Fig. 5.9 Full Subtractor

- Full subtractor can be implemented by using two half- subtractors and an OR gate.



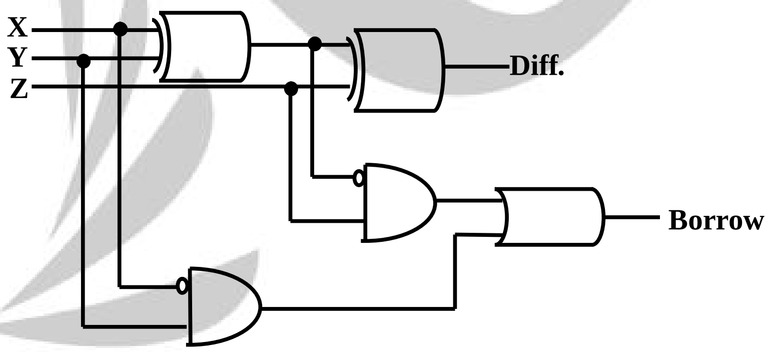


Fig. 5.10 Full Subtractor

- Full adder can be converted into full subtractor with an additional inverter.
- Four bit binary parallel adder can be constructed by using three full adders and one half adder or by using four full adders with input carry for least significant bit full adder is zero.
- Four bit binary parallel adder shown in figure is also called as Ripple carry adder.

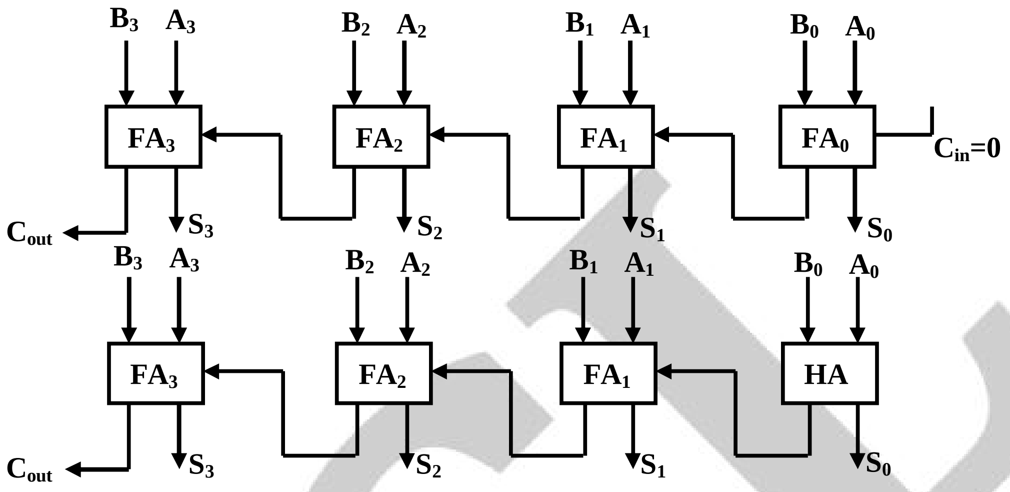


Fig. 5.11 Binary Parallel Adder

➢ Carry Look- Ahead adder is faster than ripple carry adder.

Example:

Implement Boolean function F = $ABC' + A'C + B'C$ using half adder.

Solution:

$F = AB\overline{C} + C(\overline{A} + \overline{B})$

$= AB\overline{C} + C(\overline{AB})$

$= (AB) \oplus C$

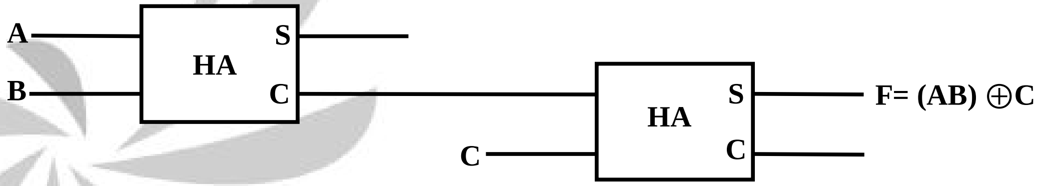


Fig. 5.12

➢ **Decoder:** A decoder is a logic circuit that converts an n bit binary input code into M ($=2^n$) output lines such that each output line will be activated for only one of the possible combinations of inputs.

(OR)

A decoder is a Combinational circuit that converts binary information from 'n' input lines to a maximum of 2^n unique output lines.

E.g. 2 × 4 line Decoder (it is also called one four line decoder)

- Decoders are available in two different types of output forms:
  (1) Active high output type decoders and
  (2) Active low output type of decoders.
- Active high output type of decoders are constructed with AND gates and active low output type of decoders are constructed with NAND gates.

Truth table of active high output type of decoder is given below:

X	Y	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

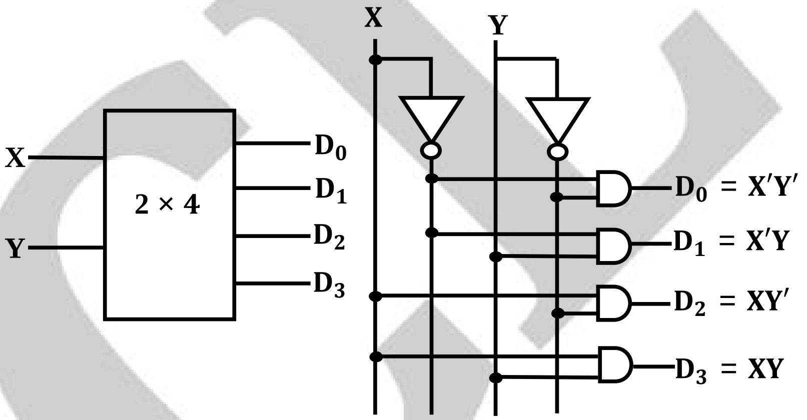


Fig. 5.13 Binary Decoder with Active High Output

- Active low output types of decoders will give the output low for given input combination and all other outputs are high.

Truth table of active low output type of decoder

X	Y	D_0	D_1	D_2	D_3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

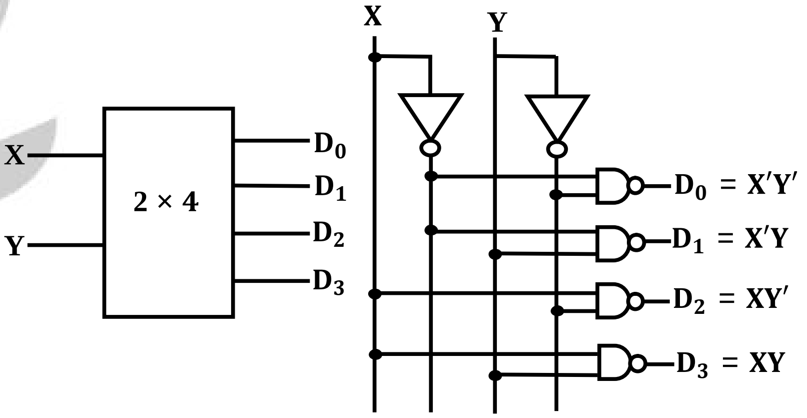


Fig. 5.14 Binary Decoder with Active Low Output

- 3 to 8 line decoder is also called Binary-to-Octal decoder or converter. It is also called 1of 8 decoder, because only one of the 8 outputs is active at a time.

- Decoders are widely used in the memory system of computer, where they respond to the address code input from the CPU to activate the memory storage location specified by the address code.
- Decoders are also used to convert binary data to a form suitable for displaying on decimal read outs.
- Decoders can be used to implement combinational circuits, Boolean functions etc.

Example:

Implement full adder with a decoder.

Solution:

$S(x, y, z) = \sum(1, 2, 4, 7)$
$C(x, y, z) = \sum(3, 5, 6, 7)$

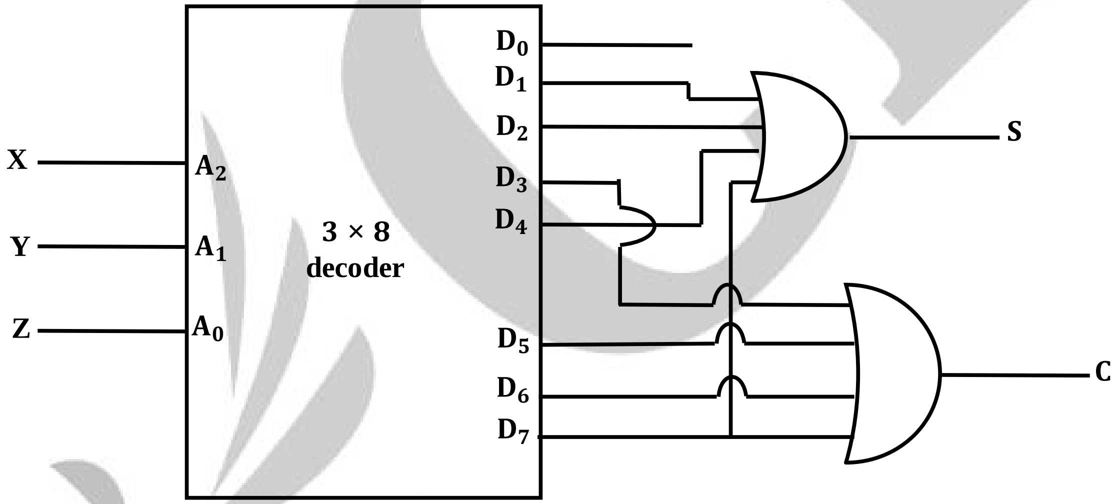


Fig 5.15

Example:
Implement 3 × 8 decoder using 2 × 4 decoder.

Solution:

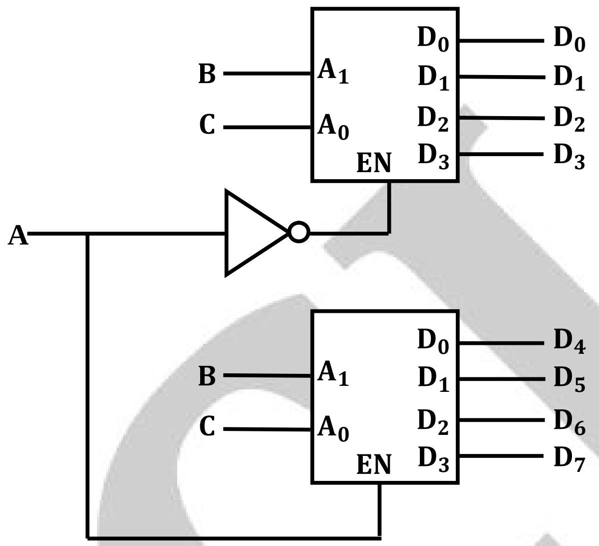


Fig 5.16

- **Demultiplexer:** A decoder with enable input, acts as demultiplexer.” "Demultiplexer", is a logical circuit that takes a single input source and sends it to one of several 2^n possible output lines. The selection of specific output line is controlled by the bit values of ‘n’ selection lines.

E	A	B	D_0	D_1	D_2	D_3
1	X	X	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	1	0	0
0	1	0	0	0	1	0
0	1	1	0	0	0	1

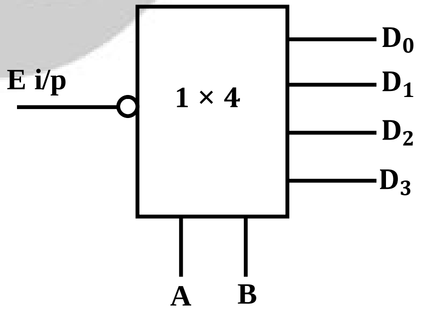
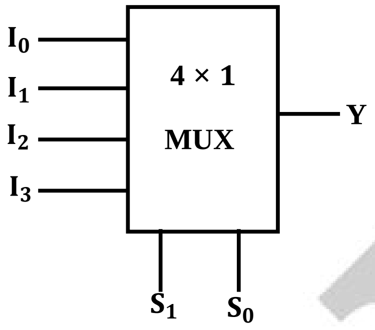


Fig. 5.17 Binary Demultiplexer

5.5 Multiplexer

A **multiplexer** or **mux** is a device that selects one of several analog or digital input signals and forwards the selected input into a single line. A multiplexer of 2^n inputs has n select lines, which are used to select which input line to send to the output.”



Truth table

S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

Fig. 5.18 Binary Multiplexer

- Multiplexers can be used for the implementation of Boolean functions, combinational circuits. They can also used for parallel to serial conversion.
- Multiplexer is also called data selector or universal circuit.
- All three variable Boolean equations can be implemented by using 8 × 1 multiplexer without using any additional gates. Some but not all three variable Boolean equations can also be implemented with 4×1 mux without using any additional gates.

Example:

Implement Boolean function F(A, B, C) = $\sum(1,3,5,6)$ with 4 × 1 mux

Solution:

Truth table

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

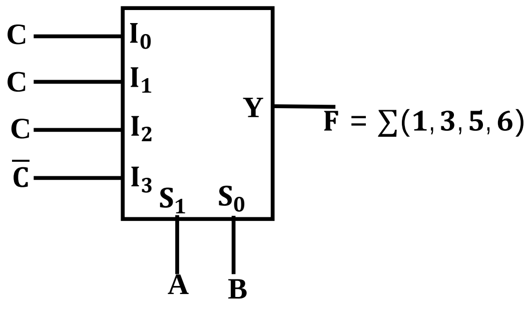


Fig 5.19

When AB = 00, F is equal to C, similarly for all other combinations of AB, input to mux is defined in terms of C.

- **Encoder:** A decoder identifies a particular code present at the input terminals of the circuit. The inverse process is called Encoding."An Encoder has number of inputs (2^n) one and only one of which is in the high state or active, and an n-bit code is generated upon which of the inputs is excited. An encoder is a digital function that produces a reverse operation from that of a decoder. An encoder has (2^n or less) input lines and n output lines. The output lines generate the binary code for the input variables. An example of an encoder shown in fig. The octal to binary encoder consists of eight inputs, one for each of the digits, and three outputs that generate the corresponding binary number.

  The encoder in figure below assumes that only one input line can be equal to 1 at any time; otherwise the circuit has no meaning. Note that the circuit has eight inputs and could have =256 possible input combinations. Only eight of these combinations have any meaning. The other inputs combinations are don't care conditions.

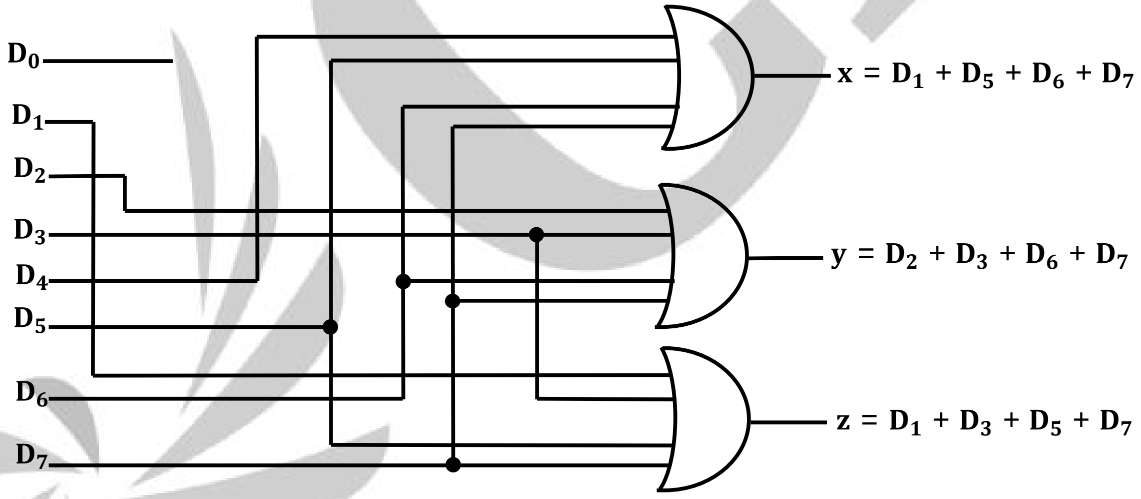

D_0
D_1
D_2
D_3
D_4
D_5
D_6
D_7
$x = D_1 + D_5 + D_6 + D_7$
$y = D_2 + D_3 + D_6 + D_7$
$z = D_1 + D_3 + D_5 + D_7$

Fig. 5.20 Octal to Binary Encoder

- **ROM (read Only Memory):** ROM is nothing but the combination of decoder and Encoder. It is a semi conductor memory and which is a permanent memory, ROM can also be defined as a Simple Code conversion unit.

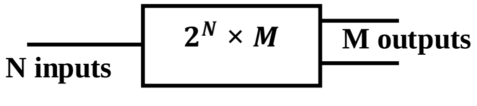


Fig. 5.21

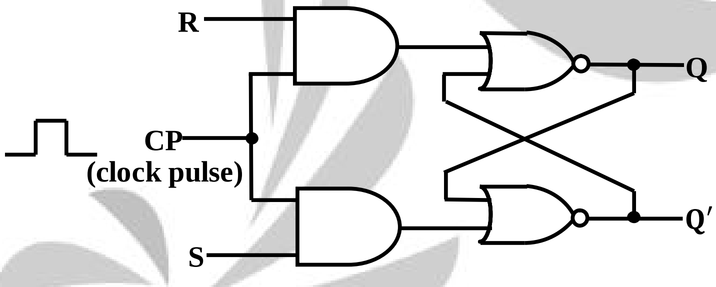
- ROM = Fixed AND, Programmable OR
- PAL = Programmable AND, Fixed OR
- PLA = Programmable AND, Programmable OR.

Important Points

- Two cross coupled inverters will form a basic latch which can store one bit of information.
- Flip-flops: Flip-Flop is also called Bistable multivibrator. It can store one bit of information.
- In a flip-flop one output is always complement of the other output.
- Flip-flop has two stable states.

5.6 Clocked S-R Flip-flop

The clocked SR flip-flop shown in Figure below consists of a basic NOR flip-flop and two AND gates. The outputs of the two AND gates remain at 0 as long as the clock pulse (or CP) is 0, regardless of the S and R input values. When the clock pulse goes to 1, information from the S and R inputs passes through to the basic flip-flop. With both S=1 and R=1, the occurrence of a clock pulse causes both outputs to momentarily go to 0. When the pulse is removed, the state of the flip-flop is indeterminate, ie., either state may result, depending on whether the set or reset input of the flip-flop remains a 1 longer than the transition to 0 at the end of the pulse.



(a) S-R Flip-flop 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) S-R Flip-flop Truth table

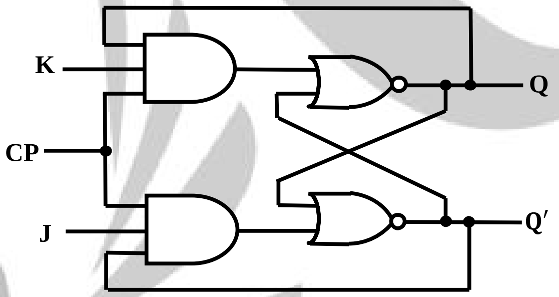
Fig. 5.22 Clocked SR flip-flop

$$Q_{n+1} = S_n + R'_n Q_n$$

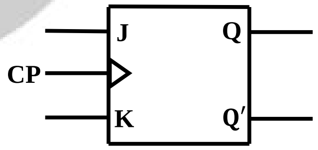
- The disadvantage of S-R flip-flop is for S=1, R=1 output cannot be determined. This can be eliminated in J-K flip-flop.
- S-R flip flop can be converted to J-K flip-flop by using the two equation S=JQ' and R= KQ.

5.7 JK-flip-flop

- A JK flip-flop is a refinement of the SR flip-flop in that the indeterminate state of the SR type is defined in the JK type. Inputs J and K behave like inputs S and R to set and clear the flip-flop (note that in a JK flip-flop, the letter J is for set and the letter K is for clear). When logic 1 inputs are applied to both J and K simultaneously, the flip-flop switches to its complement state, ie., if Q=1, it switches to Q=0 and vice versa.
- A clocked JK flip-flop is shown in figure below. Output Q is ANDed with K and CP inputs so that the flip-flop is cleared during a clock pulse only if Q was previously 1. Similarly, ouput Q' is ANDed with J and CP inputs so that the flip-flop is set with a clock pulse only if Q' was previously 1.
  Note that because of the feedback connection in the JK flip-flop, a CP signal which remains a 1 (while J=K=1) after the outputs have been complemented once will cause repeated and continuous transitions of the outputs. To avoid this, the clock pulses must have a time duration less 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. The same reasoning also applies to the T flip-flop presented next.



(a) J-K Flip-flop 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) J-K Flip-flop Transition table

Fig. 5.23 Clocked JK flip-flop

Truth table

J_n	K_n	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	Q'_n

$$Q_{n+1} = JQ'_n + K'Q_n$$

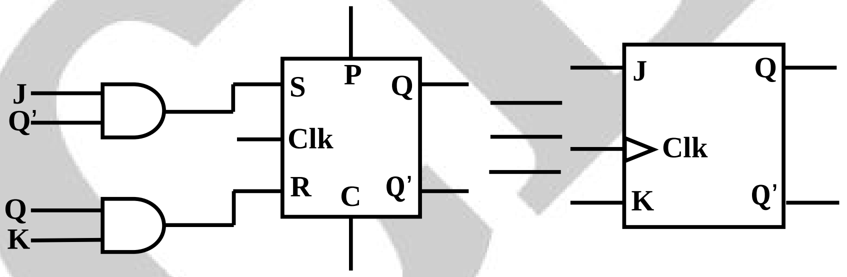
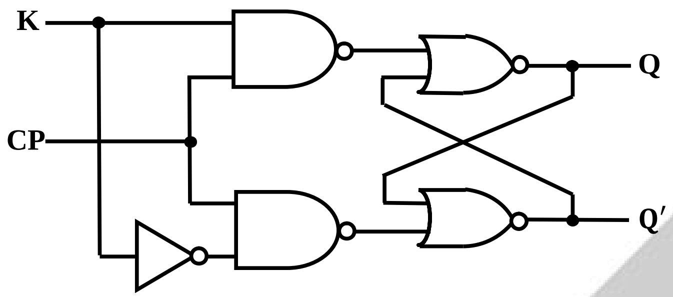


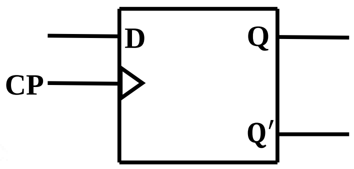
Fig 5.24

- Race around problem is present in the J-K flip flop, when both J=K=1.
- Toggling the output more than one during the clock pulse is called Race around Problem.
- The race around problem in J-K flip-flop can be eliminated by using edge triggered flip-flop or master slave J-K flip flop or by the clock signal whose pulse width is less than or equal to the propagation delay of flip-flop.
- Master-slave flip-flop is a cascading of two J-K flip-flops. Positive or direct clock pulses are applied to master and inverted clock pulses are applied to the slave flip-flop.

5.8 D-flip-flop

The D flip-flop shown in figure below is a modification of the clocked SR flip-flop. The D input goes directly into the S input and the complement of the D input goes to the R input. The D input is sampled during the occurrence of a clock pulse. If it is 1, the flip-flop is switched to the set state (unless it was already set). If it is 0, the flip-flop switches to the clear state.





(a) D Flip-flop Logic diagram with NAND gates (b) D Flip-flop Graphical symbol

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

(C) D Flip-flop Transition table

Fig. 5.25 Clocked D flip-flop

- It is also called a Delay flip-flop. By connecting an inverter in between J and K input terminals, D flip-flop is obtained. K always receives the compliment of J.

Truth table

D	Q'_{n+1}
0	0
1	1

$Q_{n+1} = D$

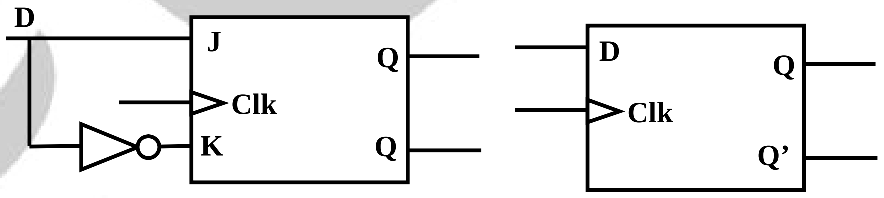
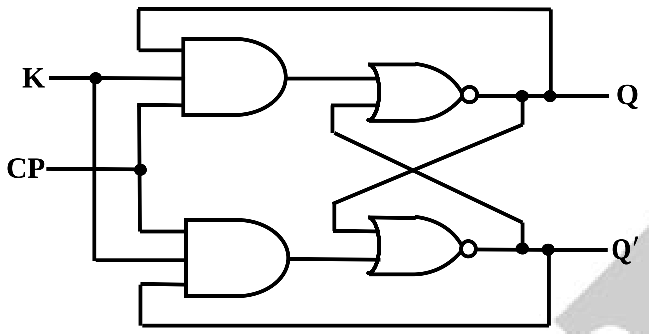


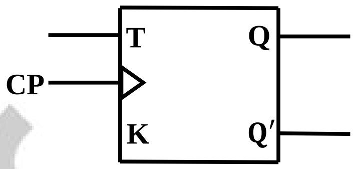
Fig. 5.26 Implementation of D Flip Flop Using JK Flip Flop

- D flip-flop is used to provide delay. The bit on the D line is transferred to the output at the next clock pulse.

5.9 T Flip-flop

The T flip-flop is a single input version of the JK flip-flop. As shown in figure below, the T flip-flop is obtained from the JK type if both inputs are tied together. The output of the T flip-flop "toggles" with each clock pulse.





(a) T Flip Flop Logic diagram **(b) T Flip Flop Graphical symbol**

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

(C) T Flip-flop Transition table

Fig. 5.27 Clocked T flip-flop

Truth table

T	Q_{n+1}
0	Q_n
1	$\overline{Q_n}$

$Q_{n+1} = TQ_n' + T'Q_n = T \oplus Q_n$

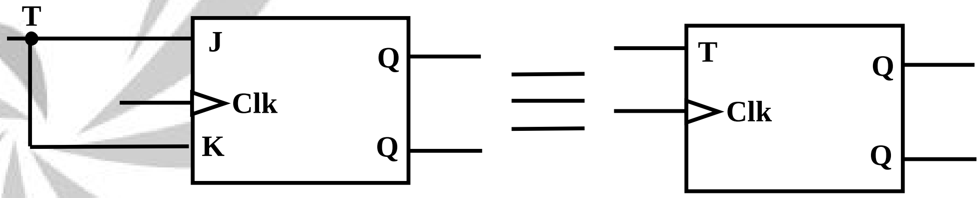


Fig. 5.28 Implementation of T Flip Flop Using JK Flip Flop

- If X KHz clock signal is applied to a T flip flop when T=1, then the output (Q) signal frequency is given by X/2KHz. Thus it acts as a frequency divider.

Setup Time (t_s): Time interval immediately preceding the active transition of clock signal during which the control input must be maintained at the proper level.

Hold Time (t_H): The time interval immediately following the active transition of the clock signal during which the synchronous control input must be maintained at the proper level.

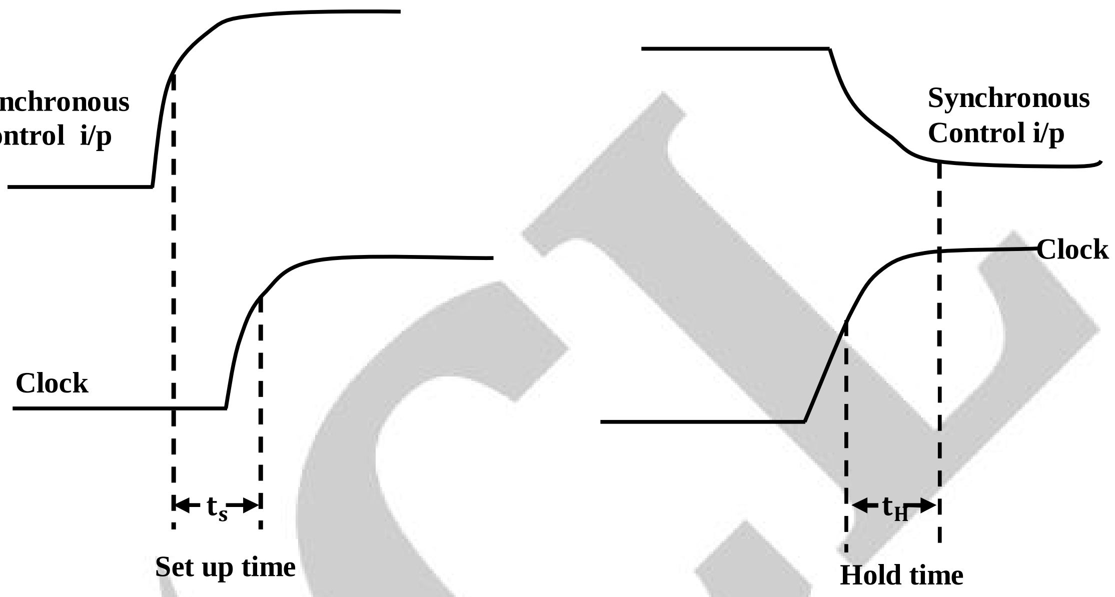


Fig. 5.29 Setup & Hold Time Representation

5.10 Registers and shifts registers:

- A register is a group of flip-flops used to store binary information. An n-bit register can store n-bit information
- A register which is able to shift the information either from left to right or from right to left is called a shift register.

∗ Shift register can perform four different operations.

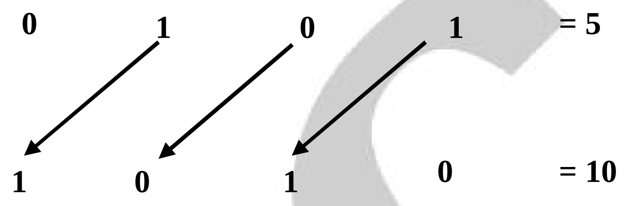
1. Serial input	Parallel output.
2. Serial input	Serial output
3. Parallel input	Parallel output
4. Parallel input	Serial output.

∗ **Universal Shift Register:** A register which is able to shift the information from left to right or from right to left and which can perform all four operations is called universal shift register.

∗ Applications of Shift registers:

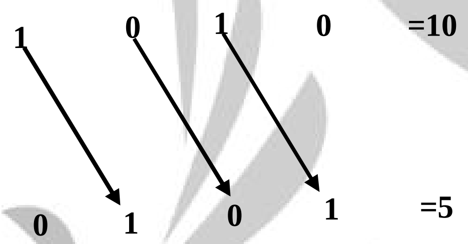
1. Serial to parallel conversion (It is also called spatial to temporal code conversion).
2. Parallel to serial conversion (It is also called temporal to spatial code conversion).
3. Sequence generator.
4. Multiplication and Division.
5. Ring counter and twisted ring counter.
6. Digital delay line (serial input and serial output operations).

→ Left shift operation is nothing but multiplied by 2.
Eg: Q_3 Q_2 Q_1 Q_0

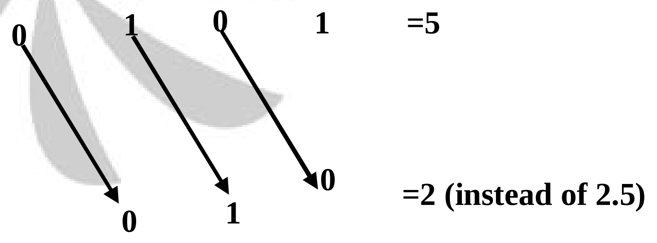


Shift left by n – positions is equivalent to multiplication by 2^n.

→ If least significant bit = 0, then right shift operation by one position is same as Division by 2.
Eg: Q_3 Q_2 Q_1 Q_0



→ If L.S.B = 1, then right shift operation gives integer division by 2.
Eg: Q_3 Q_2 Q_1 Q_0



➢ **Ring Counter:** Shift register can be used as ring counter when Q_0 output terminal is connected to serial input terminal.

- An n-bit ring counter can have "n" different output states. It can count n-clock pulses.
- Ring Counter

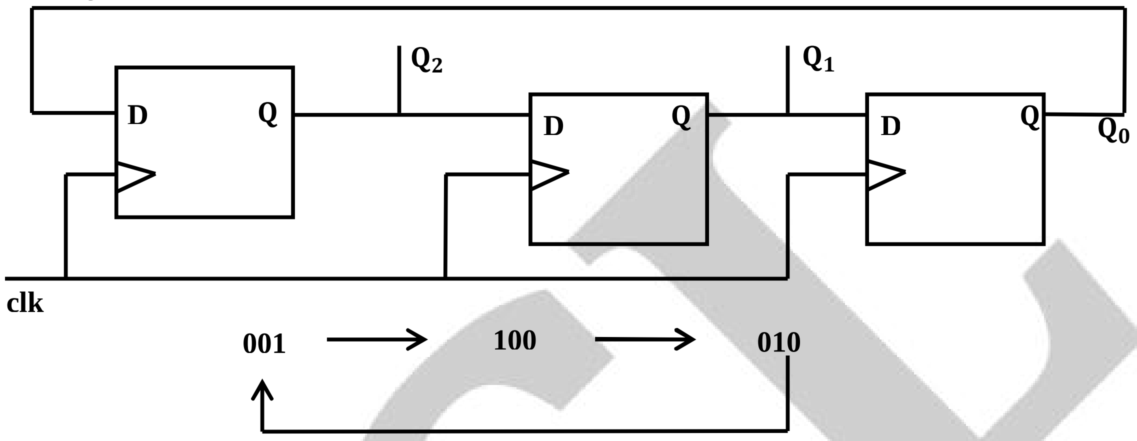


Fig. 5.30 Mod – 3, divide by 3 counter, N : 1 counter

- **Twisted Ring counter:** It is also called Johnson's Ring counter. It is formed when Q_0' output terminal is connected to the serial input terminal of the shift register.
- An n-bit twisted ring counter can have maximum of 2n different output states.
- Twisted Ring Counter is also called Johnson counter.

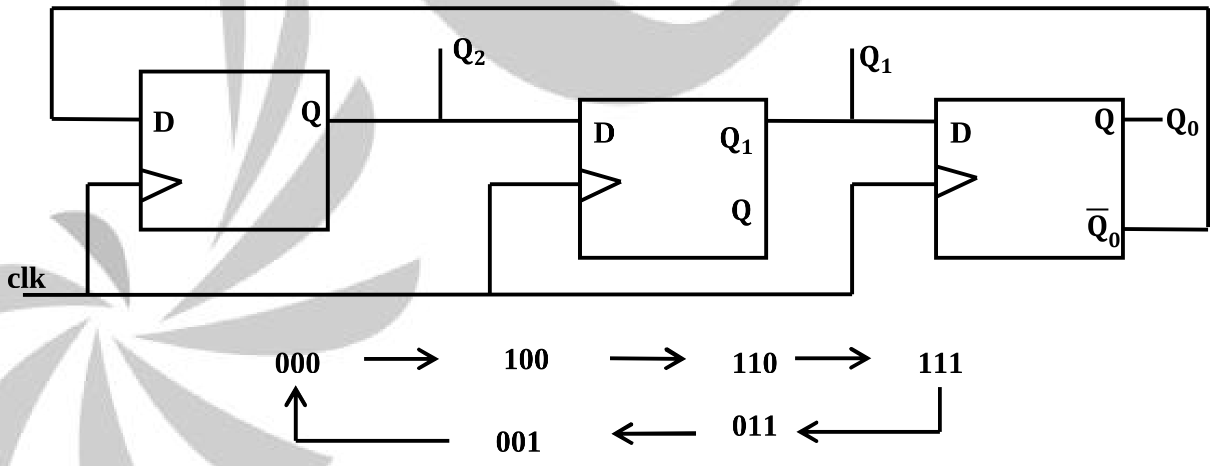


Fig. 5.31 Mod – 6, divide by 6, 2N: 1 counter

5.11 COUNTERS

→ The Counter is driven by a clock signal and can be used to count the number of clock cycles. Counter is nothing but a frequency divider circuit.

→ Two types of counters are available:
1. Synchronous. 2. Asynchronous.

→ Synchronous counters are also called parallel counters. In this type clock pulses are applied simultaneously to all the flip-flops.

→ Asynchronous counters are also called Ripple or serial counters. In this type of counters the output of one flip-flop is connected to the clock input of next flip-flop and so on.

→ **Ripple counter (Asynchronous)**

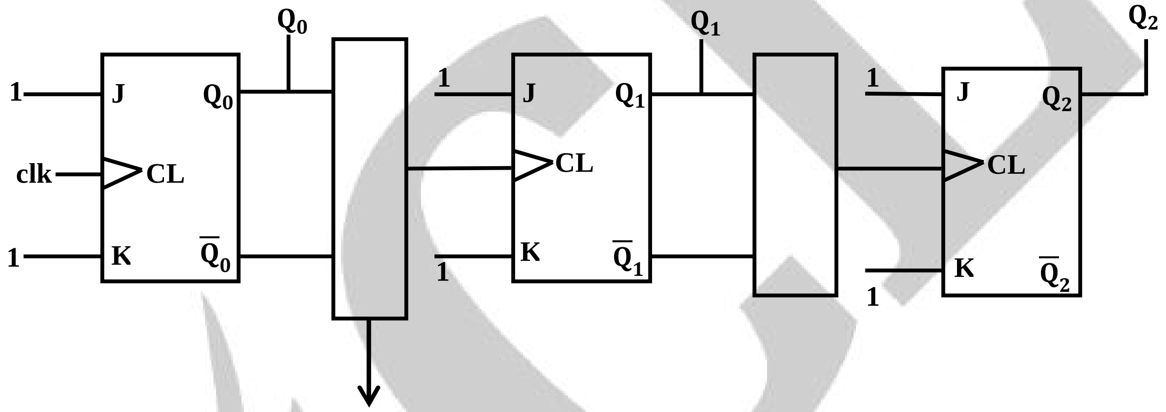


Fig. 5.32 Ripple counter

Combinational circuit which decides whether up counter or down counter

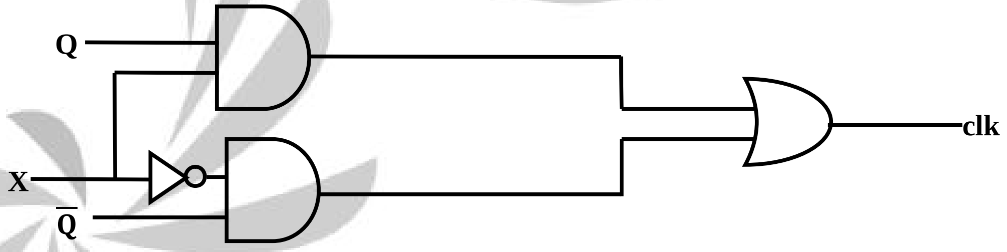


Fig. 5.33

For up – counter, X = 1 So, clk = Q

For down – counter, X = 0 So, clk = $\overline{Q}$

→ A counter having n-flip-flops can have 2^n output states i.e. it can count 2^n clock pulses (0 to $2^n - 1$).

→ The largest binary number that can be represented by an n-bit counter has a decimal equivalent of (2^n - 1). Example. : n = 3, then $2^n - 1 = 2^3 - 1 = 7$.

- → A counter can be made to count either in the up mode or in the down mode.
- → Synchronous counters are faster than asynchronous counters.
- → The modulus of a counter is the total number of states through which the counter can progress. For example mod-8 counter is having 8 different states (000 to 111).
- → The output signal frequency of Mod-n counter is 1/n th of the input clock frequency.
- → Hence that counter is also called divide by n counter.
- → The number of flip-flops (n) required to construct Mod N counter can be obtained from the following formula:
  $2^{n-1} < N \leq 2^{n}$.
- → A decade counter is also called Mod- 10 or divide by 10 counter requires 4 flip-flops.
- → Any binary counter can be a modulus counter where as the modulus counter need not be a binary counter.
- → Six flip-flops are required to construct mod-60 counter.
- → Two types of synchronous counters are available.
  1. Series carry
  2. Parallel carry.