UNIT 4 PRINCIPLE OF LOGIC CIRCUITS II

Structure **Page Nos.**

4.0 INTRODUCTION

By now you are aware of the basic configuration of computer systems, how the data is represented in computer systems, logic gates and combinational circuits. In this unit you will learn how all the computations are performed inside the system. You will come across terms like flip flops, registers, counters, sequential circuits etc. Here, you will also learn how to make circuits using combinational and sequential circuits. These circuit design will help you in performing practicals in MCSL-017 lab course.

4.1 OBJECTIVES

After going through this unit you will be able to:

- define the flip-flops and latch;
- describe behaviour of various flip-flops;
- define significance of excitation tables and state diagrams;
- define some of the useful circuits of a computer system like registers counters etc.; and
- construct logic circuits involving sequential and combinational circuits.

4.2 SEQUENTIAL CIRCUITS: THE DEFINITION

A sequential circuit is an interconnection of combinational circuits and storage elements. The storage elements, called flip-flops, store binary information that indicates the state of sequential circuit at that time. The block diagram of a sequential circuit is shown in figure 4.1.

As shown in the diagram, the present output depends upon the past Input states.

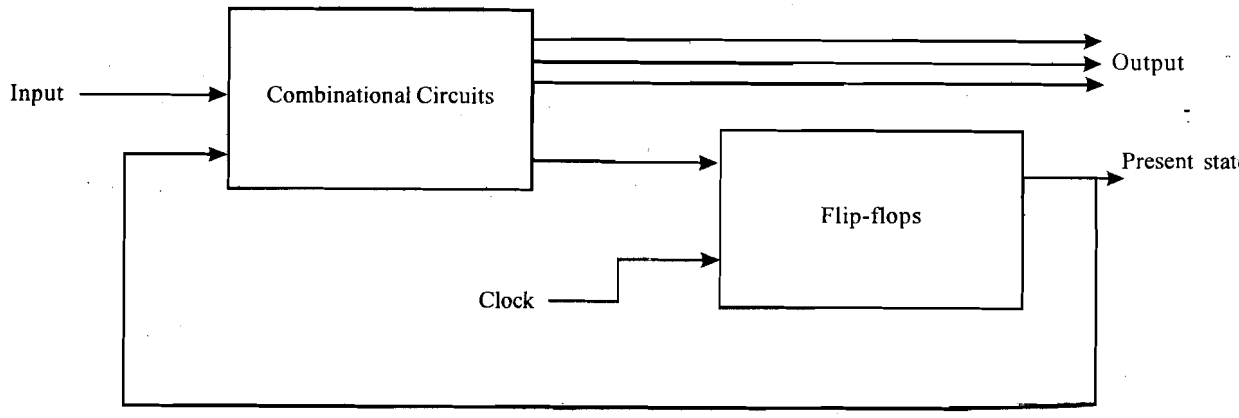


Figure 4.1: Block Diagram of sequential circuits.

These sequential circuits unlike combinational circuits are time dependent. The sequential circuits are broadly classified, depending upon the time at which these are observed and their internal state changes. The two broad classifications of sequential circuits are:

- Synchronous
- Asynchronous

Synchronous circuits use flip-flops and their status can change only at discrete intervals (Doesn't it seems as good choice for discrete digital devices such a computers?). Asynchronous sequential circuits may be regarded as combinational circuit with feedback path. Since the propagation delays of output to input are small, they may tend to become unstable at times Thus, complex asynchronous circuits are difficult to design.

The synchronization in a sequential circuit is achieved by a clock pulse generator, which gives continuous clock pulse. Figure. 4.2. shows the form of a clock pulse.

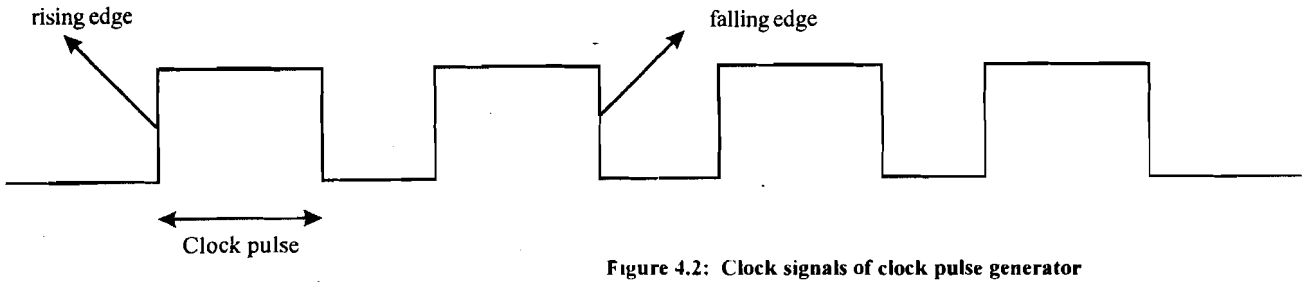


Figure 4.2: Clock signals of clock pulse generator

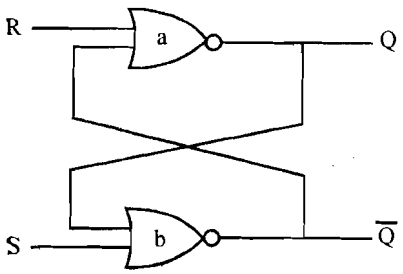
A clock pulse can have two states: - 0 or 1; disabled or active state. The storage elements can change their state only when a clock pulse occurs. Sequential circuits that have clock pulses as input to flip-flops are called clocked sequential circuit.

4.3 FLIP-FLOPS

Let us see flip-flops in detail. A flip-flop is a binary cell, which stores 1-bit of information. It itself is a sequential circuit. By now we know that flip-flop can change its state when clock pulse occurs but when? Generally, a flip-flop can change its state when the clocks transitions from 0 to 1 (rising edge) or from 1 to 0 (falling edge) and not when clock is 1. If the storage element changes its state when clock is exactly at 1 then it is called ***latch***. In simple words, **flip-flop** is ***edge-triggered*** *and* latch is ***level-triggered.***

4.3.1 Basic Flip-flops

Let us first see a basic **latch**. A latch or a flip-flop can be constructed using two NOR or NAND gates. Figure 4.3 (a) shows logic diagram for S-R latch using NOR gates. The latch has two inputs S & R for set and reset respectively. When the output is Q=1 & $\overline{Q}$ =0, the latch is said to be in the set state. When Q=0 & $\overline{Q}$ =1, it is the reset state. Normally, The outputs Q & $\overline{Q}$ are complement of each other. When both inputs are equal to 1 at the same time, an undefined state results, as both outputs are equal to 0.



(a) Logic Diagram

S	R	Q	$\overline{Q}$	
1	0	1	0	
0	0	1	0	Set State
0	1	0	1	
0	0	0	1	Reset State
1	1	0	0	Undefined

(b) Truth Table

Figure. 4.3: SR Latch using NOR gates

Figure 4.3 (b) Shows truth table for S-R latch. Let us examine the latch more closely.

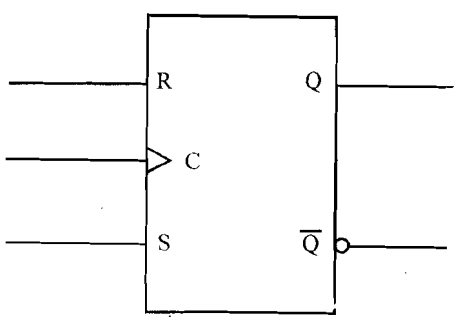
i) Say, initially 1 is applied to S leaving R to 0 at this time. As soon as S=1, the output of NOR gate 'b' goes to 0 i.e. $\overline{Q}$ becomes 0 and almost immediately Q becomes 1 as both the inputs ($\overline{Q}$ & R) to NOR gate 'a' become 0. The change in the value of S back to 0 does not change $\overline{Q}$ as the input to NOR gate 'b' now are $\overline{Q}$ = 1 & S=0. Thus, the flip-flop stays in set state even after S returns to 0.

ii) If R goes to 1 then latch acquires clear state. On changing R to 1, Q changes to 0 irrespective of the state of flip-flop and as Q is 0 & S is 0 then $\overline{Q}$ becomes 1. Even after R returns to 0, Q remains 0 i.e. latch is in clear state.

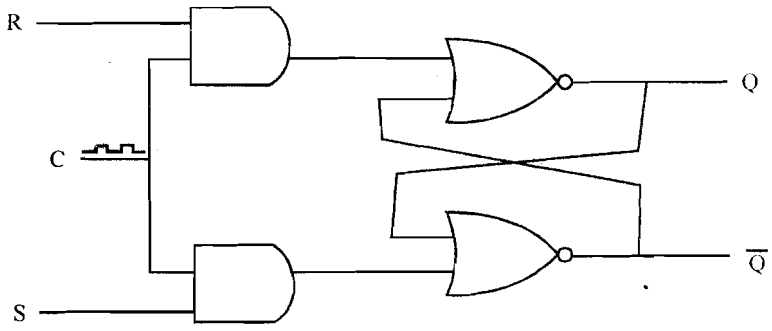
iii) When both S & R go to 1 simultaneously, the two outputs go to 0. This gives undefined state.

Let us try to construct most common flip- flops from this basic latch.

R-S Flip flop - The graphic symbol of S-R flip-flop is shown in Fig 4.4. It has three inputs, S (set), R (reset) and C (for clock). The Q(t+1) is the next state of flip-flop after the occurrence of a clock pulse. Q(t) is the present state, that is present Q value (Set-1 or Reset– 0).



(a) Graphic Symbol



(b) Logic Diagram

C	S	R	Q(t+1)
0	0	0	Q(t) No change in state
1	0	1	0 Reset
1	1	0	1 Set
1	1	1	x undefined

(c) Characteristic Table

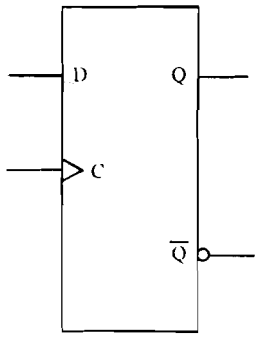
Figure 4.4: R-S Flip-flop

In figure 4.4 (a), the arrowhead symbol in front of clock pulse C indicates that the flip-flop responds to leading edge (from 0 to 1) of input clock signal.
Operation of R-S flip-flop can be summarised as:

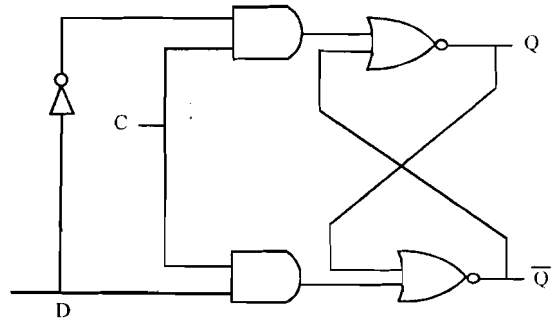
1) If no clock signal i.e. C=0 then output can not change irrespective of R & S values
2) When clock signal changes from 0 to 1 and S=1, R=0 then output Q=1 & $\overline{Q}$=0 (Set)
3) If R=1 S=0 & clock signal C changes from 0 to 1 then output Q=0 & $\overline{Q}$=1 (Reset)
4) During positive clock transition if both S & R become 1 then output is not defined, as it may become 0 or 1 depending upon internal timing delays occurring in circuit.

D Flip -Flop

The D (data) flip-flop is modification of RS flip-flop. The problem of undefined output in SR flip-flop when both R & S become 1 gets avoided in D flip-flop. The simple solution to avoid such condition is by providing just a single input. Thus, the non-clocked inputs to AND gates (S &R of fig 4.4 (b)) are guaranteed to be opposite of each other by inserting an **inverter** between them. The logic diagram and characteristic table of D flip flop is shown in figure 4.5.



(a) Graphic Symbol



(b) Logic Diagram

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

(c) Characteristic Table

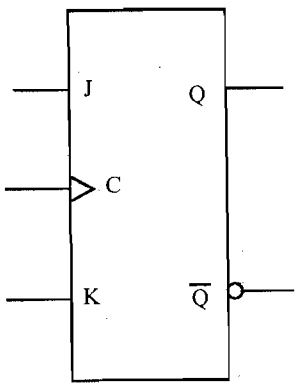
Figure 4.5: D Flip flop

D flip-flop is also referred as Delay flip-flop because it delays the 0 or 1 applied to its input by a single clock pulse.

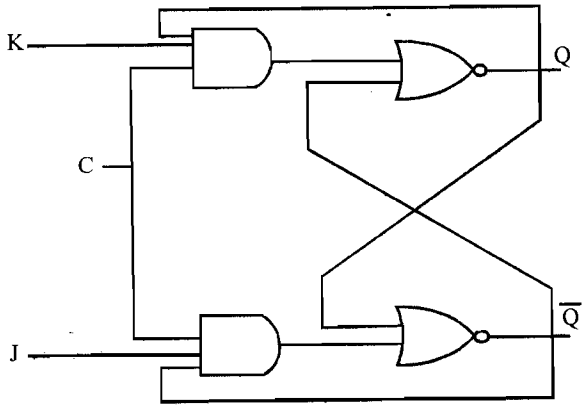
J-K flip-flop

The J-K flip-flop is also a modification of SR flip-flop, it has 2 inputs like S & R and all possible inputs combinations are valid in J K flip-flop.

Figure. 4.6 shows implementation of J K flip-flop. The inputs J & K behave exactly like input S & R to set and reset flip-flop, respectively. When J & K are 1, the flip-flop output is complemented with clock transition. [Try this as an exercise]



(a) Graphic Symbol



(b) Logic Diagram

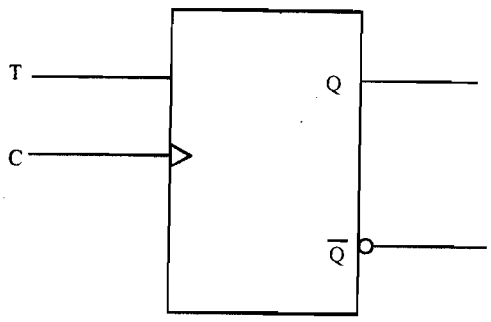
J	K	Q(t+1)
0	0	Q(t) No Change
0	1	0 Clear
1	0	1 Set
1	1	$\overline{Q}$(t) Complement

(c) Characteristic Table

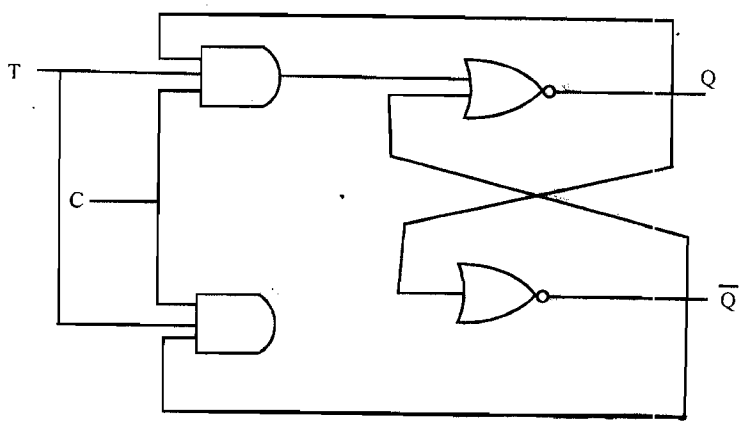
Figure 4.6: J – K Flip flop

T flip-flop

T (Toggle) flip-flop is obtained from JK flip-flop by joining inputs J &K together. The implementation of T flip-flop is shown in figure. 4.7. When T=0, the clock pulse transition does not change the state. When T=1, the clock pulse transition complement the state of the flip-flop.



(a) Graphic Symbol



b) Logic Diagram

T	Q(t+1)
0	Q(t) No Change
1	$\overline{Q}$(t) Complement

(c) Characteristic Table

Figure 4.7: T- Flip flop

4.3.2 Excitation Tables

The characteristic tables of flip-flops provide the next state when inputs and the present state are known. These tables are useful for analysis of sequential circuits. But, during the design process, we know the required transition from present state to next state and wish to find the required flip-flop inputs. Thus comes the need of a table that lists the required input values for given change of state. Such a table is called excitation Table. Fig 4.8 shows excitation tables for all flip-flops.

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

(a) J-K Flip flop

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

(b) S-R flip flop

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

(c) D Flip flop

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

(d) T Flip flop

Figure 4.8: Excitation Tables for flip-flops

Q(t) & Q(t+1) indicates present and next state for flip a flop, respectively. The symbol X in the table means don't care condition i.e. doesn't matter whether input is 0 or 1.

Let us discuss more deeply, how these excitation tables are formed. For this, we take an example of J-K Flip flop.

1) The state transition from present state 0 to next state 0 (Figure 408 (a) can be achieved when

   (a) J=0, K=0, then no change in the state of flip flop

   (b) J=0, K=1, then flip flop resets i.e. 0

   (remember J-K Characterstic table from figure 4.6)

   Thus in either case J=0 but K can be 0 or 1 that is represented by don't care condition X.

2) The state transition from present state 0 to next state 1 can be achieved when

   (a) J=1, K=0, then flip flop is set i.e. 1

   (b) J=1, K=1, then flip flop is complemented i.e.change from 0 to 1

   Here, also in either case J=1 but K can be 0 orl that means again K is represented as a don't care case.

3) Similarly, state transition from present state 1 to next state 0 can be achieved when

   (a) J=0, K=1, flip flop is reset i.e.0

   (b) J=1, K=1, flip flop is complemented i.e. changes from 1 to 0

This indicates that in either case K=1 but J can be either 0 or 1 thus don't care case.

4) For state transition from present state 1 to next state 1 can be achieved when

(a) J=0, K=0, no change in flip flop

(b) J=1, K=0, flip flop is set i.e 1

Thus J is don't care case but K=0.

This whole process can be summarized in the table below:

Present State	**Next State**	**Can be achieved**
0	0	a) J=0, K=0, since Q (t) =0
		b) J=0, K=1, flip flop resets
0	1	a) J=1, K=0, flip flop set
		b) J=1, K=1, flip flop complements, Q (t) =0=Q (t+1) =1
1	0	a) J=0, K=1, flip flop reset
		b) J=1, K=1, complement Q (t)= $\overline{Q(t)}$
1	1	a) J=0, K=0, no change
		b) J=1, K=0, flip – flop set

Similarly, the excitation tables for the rest of the flip-flops can be derived (Try to do this as an exercise).

Check Your Progress 1

1. What are sequential circuits?

.....

.....

.....

.....

2. What is flip- flop? How is different from latch?

.....

.....

.....

.....

3. What is the difference between excitation table and characteristic table?

.....

.....

.....

4.3.3 Master-Slave Flip-Flop

The master slave flip-flop consists of two flip-flops. One is the master flip-flop & other is called the slave flip-flop. Fig 4.9 shows implementation of master-slave flip-flop using J-K flip-flop.

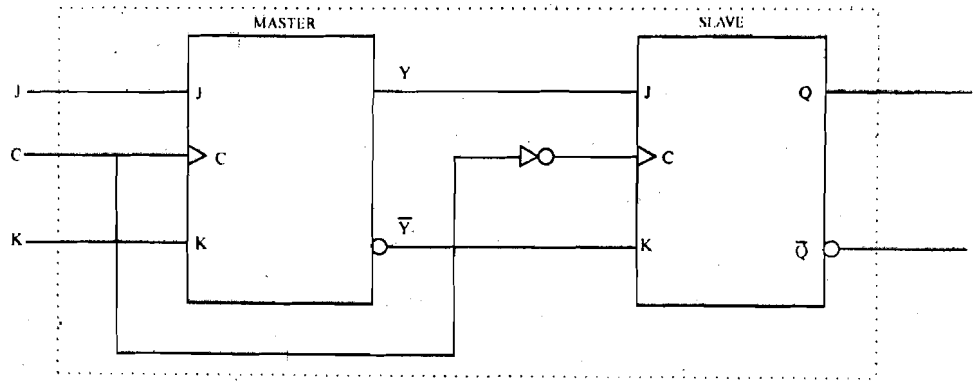


Figure 4.9: Master – Slave flip- flop

Note: Master-slave flip-flop can be constructed using D or SR flip-flop in the same manner.

Now, let us summarize the working of this flip-flop:

(i) When the clock pulse is 0, the master flip-flop is disabled but the slave becomes active and it's output Q & $\overline{Q}$ becomes equal to Y and $\overline{Y}$ respectively. Why? Well the possible combination of the value of Y and Y' are either Y=1, $\overline{Y}$ =0 or Y=0 $\overline{Y}$ =1. Thus, the slave flip-flop can have following combinations: -

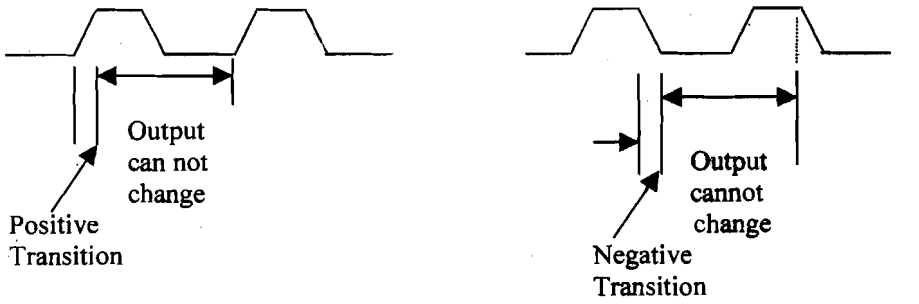
(a) J=1, K=0	which means Q=1, $\overline{Q}$ =0	(stet flip-flop)
(b) J=0, K=1	which means Q=0, $\overline{Q}$ =1	(clear flip-flop)

(ii) When inputs are applied at JK and clock pulse becomes 1, only master gets activated resulting in intermediate output Y going to state 0 or 1 depending on the input and previous state. Remember that during this time slave is also maintaining its previous state only. As the clock pulse becomes 0, the master becomes inactive and slave acquires the same state as master as explained in (a) and (b) conditions above.

But why do we require this master-slave combination? To understand this, consider a situation where output of one flip-flop is going to be input of other flip-flop. Here, the assumption is that clock pulse inputs of all flip-flops are synchronized and occur at the same time. The change of state of master occurs when the clock pulse goes to 1 but during that time the output of slave still has not changed, thus the state of the flip-flops in the system can be changed simultaneously during the same clock pulse even though output of flip-flops are connected to the inputs of other flip-flops.

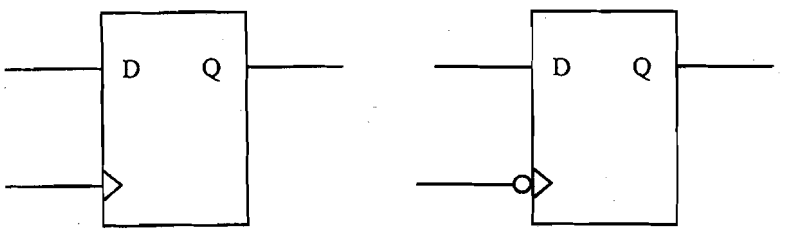
4.3.4 Edge-Triggered flip-flops

An edge-triggered flip-flop is used to synchronize the state change during a clock pulse transition instead of constant level. Some edge-triggered flip-flops trigger on the rising edge (0 to 1 transition) whereas others trigger on the falling edge (1- to 0 transition). Fig 4.10 shows the clock pulse signal in positive & negative edge-triggered flip-flops.



(a) Pulse in positive edge-triggered flip-flop **(b) Pulse in Negative edge-triggered flip flop**

Figure. 4.10: Pulse signal



(a) Positive edge-triggered D flip-flop **(b) Negative edge-triggered D flip-flop**

Figure 4.11: Edge triggered D flip- flops

The effective positive clock transition includes a minimum time called **setup time,** for which the D input must be maintained at constant value before the occurrence of clock transition. Similarly, a minimum time called **hold time,** for which the D input must not change after the application of positive transition of the pulse.

Check Your Progress 2

1. What are the advantages of master- slave flip-flop?

   .....

   .....

2. What are edge- triggered flip-flops?

   .....

   .....

4.4 SEQUENTIAL CIRCUIT DESIGN

A sequential circuit is specified by a time sequence of external inputs, external outputs and internal flip-flop binary states. Thus firstly, a *state table and state diagram* is used to describe behaviour of the circuit. Then from the state table, we get information for making logic circuit diagram.

Let us first see what is state table and state diagram. A **state table** includes the functional relationships between the inputs, output and flip-flop states (present and next) of a sequential circuit. A state diagram pictorially describes the state transition. In state diagram, a circle describes a state and directed lines indicate the transition between states. The state of flip-flop is written inside the circle. The directed lines are labelled with two binary numbers separated by a slash. The first one indicates the input value during present state and second number indicates output during present state. The state diagram of a binary counter is given in figure 4.12 (b).

The following is the procedure for design of sequential circuits:

1) Draw state table or state diagram from the problem statement, (if state diagram is available, draw state table also)
2) Give binary codes to states.
3) From state table, make input equation in simplified form. i.e. generating Boolean functions which describes signals for the inputs of flip-flops.
4) From state table, derive output equation in simplified form.
5) Draw logic diagram with required flip-flops and combinational circuits.

Let us take an example to illustrate the above procedure. Suppose we want to design 2-bit binary counter using D flip-flop. The circuit goes through repeated binary states 00, 01, 10 and 11 when external input X = 1 is applied. The state of circuit will not change when X = 0. The state table & state diagram for this is shown in figure 4.12. But how do we make this state diagram? Please note the number of flip-flops– 2 in our example as we are designing 2 bits counter. Various states of two bit input would be 00 01 10 and 11. These are shown in circle. The arrow indicate the transitions on an input value X. For example, when the counter is in state 00 and input value X=0 occurs, the counter remains in 00 state. Hence the loop back on X= 0. However, on encountering X=1 the counter moves to state 01. Like wise in all other states similar transition occur. For making state table remember the excitation table of D flip-flop given in figure 4.8 (c).

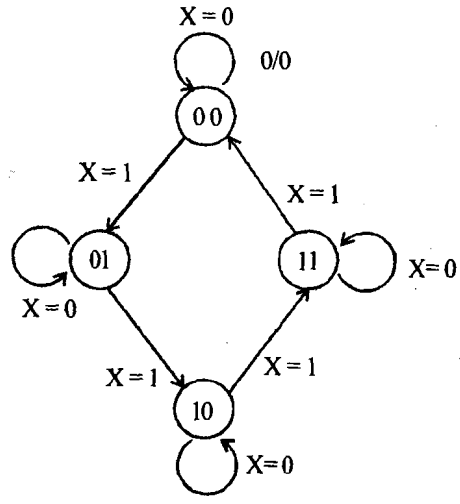
The present state of the two flip-flops and next states of the flip-flops are put into the table along with any input value. For example, if the present state of flip-flops is 01 and input value is 1 then counter will move to state 10. Notice these values in the fourth row of the values in the state table (figure 4.12 (a).
Or we can write as

A	B ⟶	A (Next)	B (Next)
0	1 X =1	1	0

This implies that flip-flop. A has moved from state clear to set. As we are making the counter using D flip-flop, the question is what would be the input D_A value of A flip-flop that allows this transition that is Q(t) = 0 to Q(t+1) =1 possible for A flip flop. On checking the excitation table for D Flip-flop, we find the value of D input of A flip-flop (called D_A in this example) would be 1. Similarly, the B flip-flop have a transition Q(t) = 1 to Q(t+1)=0, thus, D_B, would be 0. Hence notice the values of flip-flop inputs D_A and D_B. (Row 3).

Present state		Input	Next state		Flip flop Input	
A	B	X	A	B	D_A	D_B
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	0	1	0	1
0	1	1	1	0	1	0
1	0	0	1	0	1	0
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	0	0	0	0

(a) State Table



(b) State Diagram

Figure 4.12: Binary Counter Design

Next step indicates simplification of input equation to flip-flop which is done using K-Maps as shown in fig 4.13. But why did we make K-map for D_A or D_B which happens to be flip-flop input values? Please note in sequential circuit design, we are

designing the combinational logic that controls the state transition of flip-flops. Thus, each input to a flip-flop is one output of this combinational logic and the present state of flip-flops and any other input value form the input values to this combinational logic.

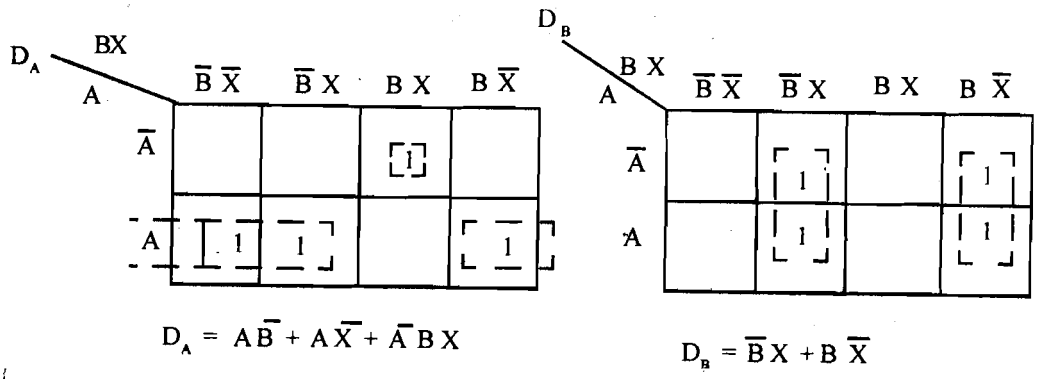


Figure 4.13: Maps for combinational circuit of 2-bit counter

Thus, two simplified flip–flop input equations are derived:

$$D_A = A\overline{B} + A\overline{X} + \overline{A}\,BX$$

$$D_B = \overline{B}\,X + B\overline{X}$$

The logic diagram is drawn in fig 4.14.

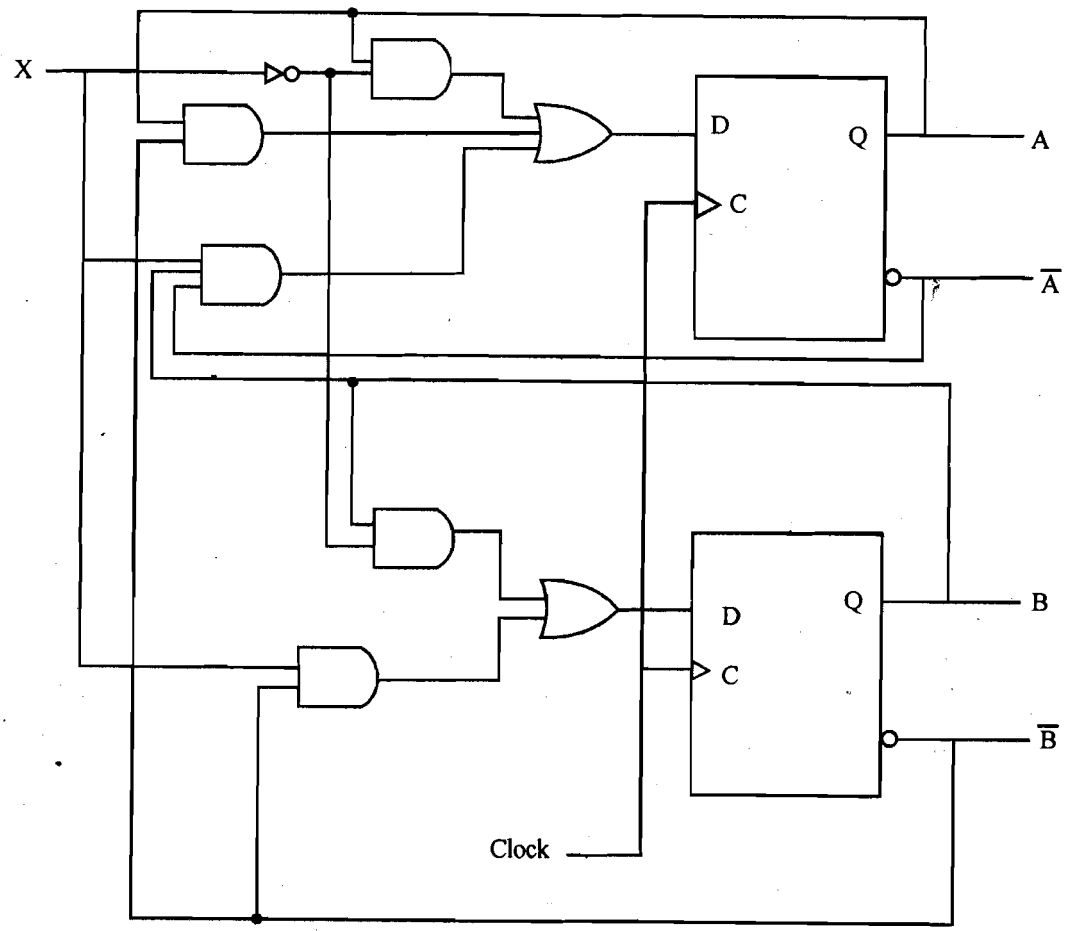


Figure 4.14: Logic diagram for 2-bit Binary Counter

Note: Similarly, the sequential circuits can be designed using any number of flip-flops using state diagrams and combinational circuits design methods.

4.5 EXAMPLES OF SEQUENTIAL CIRCUITS

Let us now discuss some of the useful examples of sequential circuits like registers, counters etc.

4.5.1 Registers

A register is a group of flip-flops, which store binary information, and gates, which controls when and how information is transferred to the register. An n-bit register has n flip-flops and stores n-bits of binary information. Two basic types of registers are: parallel registers and shift registers.

A parallel register is one of the simplest registers, consisting of a set of flip-flops that can be read or written simultaneously. Fig. 4.15 shows a 4-bit register with parallel input-output. The signal lines I_0 to I_3 inputs to flip-flops, which may be output of other arithmetic circuits like multipliers, so that data from different sources can be loaded into the register. It has one additional line called clear line, which can clears the register completely. This register is called a parallel register as all the bits of the register can be loaded in a single clock pulse.

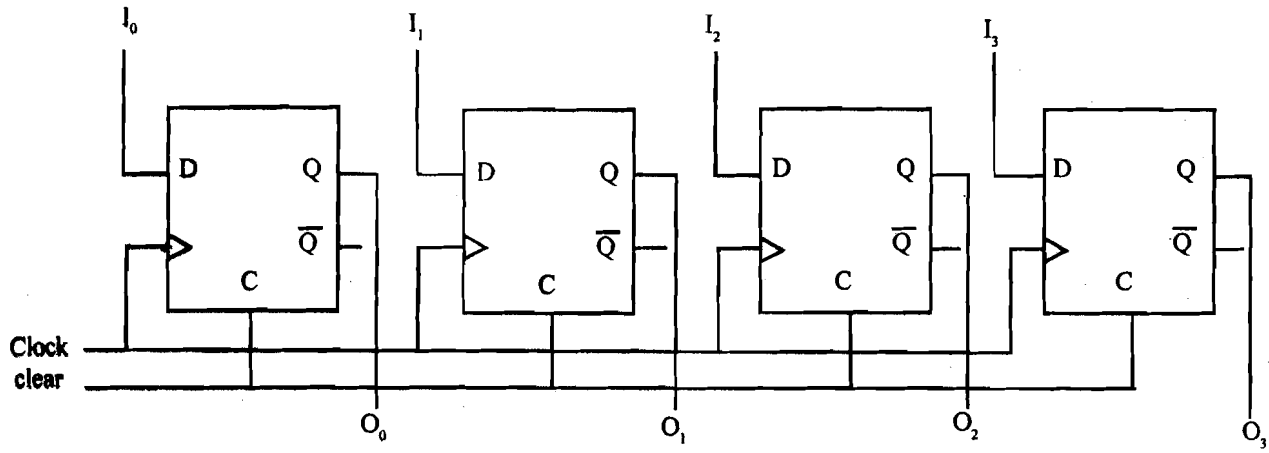


Figure 4.15: 4-bit parallel register

A shift register is used for shifting the data to the left or right. A shift register operates in serial input-output mode i.e. data is entered in the register one bit at a time from one end of the register and can be read from the other end as one bit at a time. Fig. 4.16 shows a 4-bit right shift register using D logical shift functions.

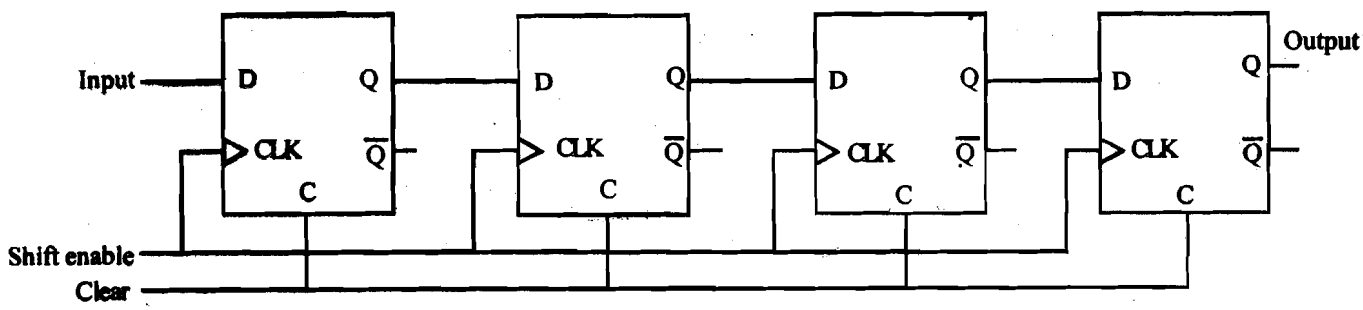


Figure 4.16: 4-bit right – shift register

Please note that in this register signal shift enable is used instead of clock pulse, why? Because it is not necessary that we want the register to perform shift on each clock pulse.

A register, which shifts data only in one direction, is called **uni-directional shift register** and a register, which can shift data in both directions, is called **bi-directional shift register**. Shift register can be constructed for bi-directional shift with parallel input-output. A general shift register structure may have parallel data transfer to or from the register along with added facility of left or right shift. This structure will require additional control lines for indicating whether parallel or serial output is desired and left or right shift is required. A general symbolic diagram is shown in Fig. 4.17 for this register.

There are 3 main control lines shown in the above figure. If parallel load enable is active, parallel input-output operation is done otherwise serial input- output shift select line for selecting right or left shift. If it has value 0 then right shift is performed and for value 1, left shift is done. Shift enable signal indicates when to start shift.

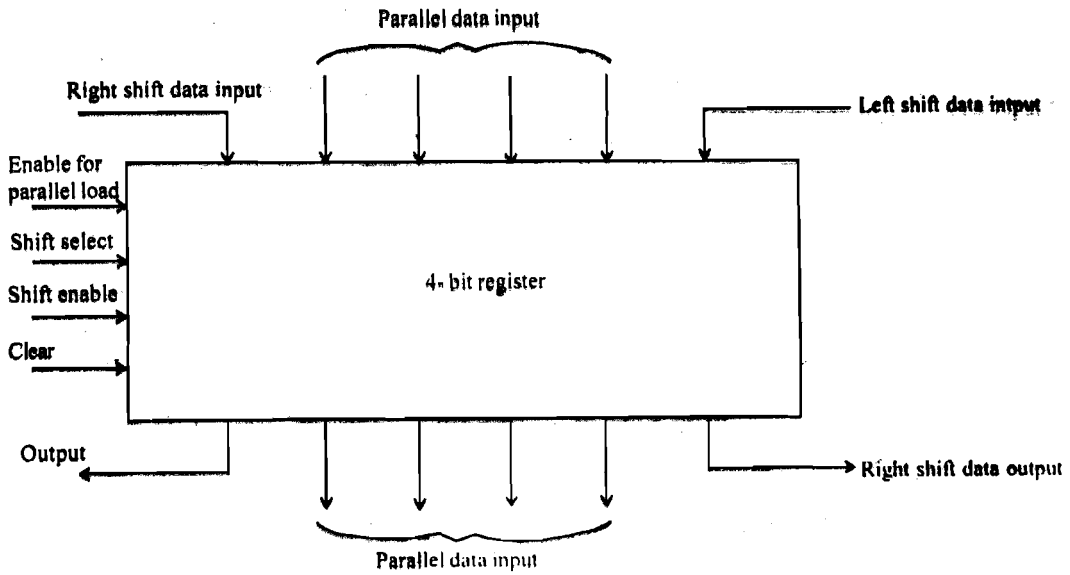


Figure. 4.17: 4 – bit left shift register with parallel load

4.5.2 Counters Asynchronous Counters

A counter is a register, which goes through a predetermined sequence of states when clock pulse is applied. In principle, the value of counters is incremented by 1 module the capacity of register i.e. when the value stored in a counter reaches its maximum value, the next incremented value becomes zero. The counters are mainly used in circuits of digital systems where sequence and control operations are performed, for example, in CPU we have program counter (PC).

Counters can be classified into two categories, based on the way they operate: Asynchronous and synchronous counters. In Asynchronous counters, the change in state of one flip-flop triggers the other flip-flops. Synchronous counters are relatively faster because the state of all flip-flops can be changed at the same time.

Asynchronous Counters : This is more often referred to as ripple counter, as the change, which occurs in order to increment the counter ripples through it from one end to the other. Fig 4.18 shows an implementation of 4-bit ripple counter using J-K flip-flops. This counter is incremented on the occurrence of each clock pulse and counts from 0000 to 1111 (i.e. 0 to 15).

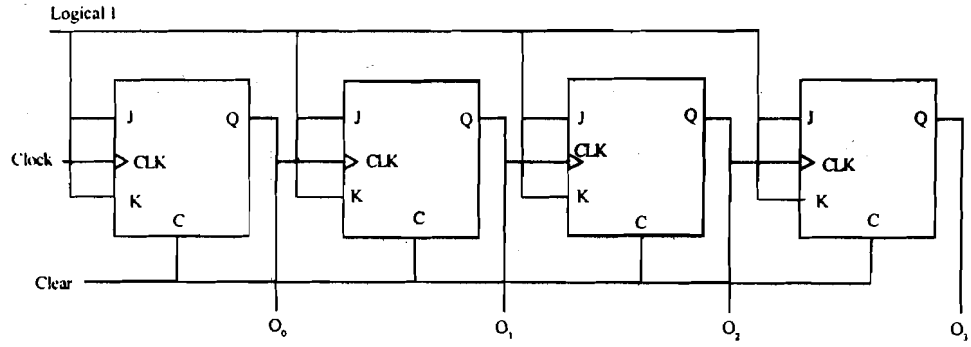


Figure. 4.18: 4 – bit ripple counter

The input line to J & K of all flip-flops is kept high i.e. logic1. Each time a clock pulse occurs the value of flip-flop is complemented (Refer to characteristic table of J K flip-flop in Figure. 4.6 (c). Please note that the clock pulse is given only to first flip-flop and second flip-flop onwards, the output of previous flip-flop is fed as clock signal. This implies that these flip-flops will be complemented if the previous flip-flop has a value 1. Thus, the effect of complement will ripple through these flip-flops.

4.5.3 Synchronous Counters

The major disadvantage of ripple counter is the delay in changing the value. How? To understand this, take an instance when the state of ripple counter is 0111. Now the next state will be 1000, which means change in the state of all flip-flops. But will it occur simultaneously in ripple counter? No, first O_0 will change then O_1, O_2 & lastly O_3. The delay is proportional to the length of the counter. Therefore, to avoid this disadvantage of ripple counters, synchronous counters are used in which all flip-flops change their states at same time. Fig 4.19 shows 3-bit synchronous counter.

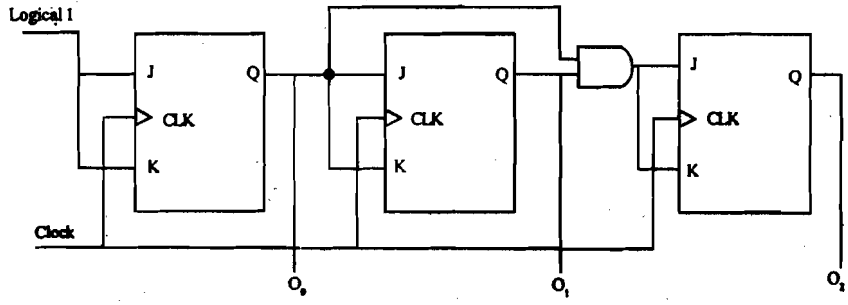


Figure 4.19: Logic diagram of 3-bit synchronous counter

You can understand the working of this counter by analyzing the sequence of states (O_0, O_1, O_2) given in Figure 4.20

O_2	O_1	O_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
0	0	0

Figure. 4.20 : Truth table for 3 bit synchronous counter

The operation can be summarized as: -

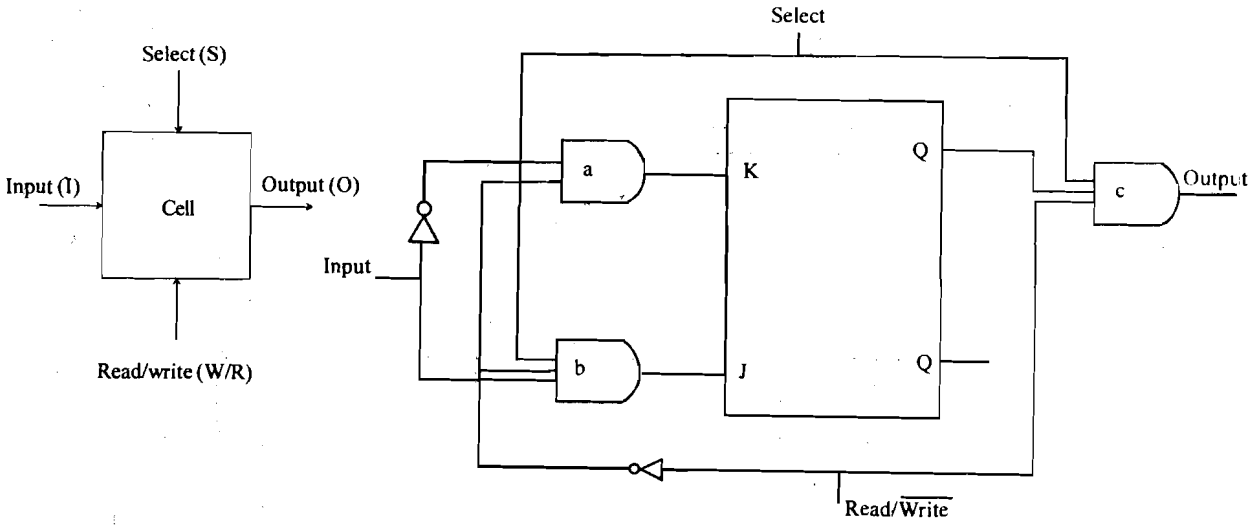
i) The first flip-flop is complemented in every clock cycle

ii) The second flip-flop is complemented on occurrence of a clock cycle if the current state of first flip-flop is 1.

iii) The third flip-flop is fed by an AND gate which is connected with output of first and second flip-flops. It will be complemented only when first & second flip-flops are in Set State.

4.5.4 RAM (Random Access Memory)

Here we will confine our discussion, in general to the RAM only as an example of sequential circuit. A memory unit is a collection of storage cells or flip flops alongwith associated circuits required to transfer information in and out of the device. The access time and cycle time it takes are constant and independent of the location, hence the name random access memory.



(a) Block Diagram **(b) Logic Diagram**

Figure 4.21: Binary Cell

RAMs are organized (logically) as words of fixed length. The memory communicates with other devices through data input and output lines, address selection lines and control lines that specify the direction of transfer.

Now, let us try to understand how data is stored in memory. The internal construction of a RAM of 'm' words and 'n' bits per word consists of m × n binary cells and associated circuits for dectecting individual words. Figure 4.21 shows logic diagram and block digram of a binary cell.

The input is fed to AND gate 'a' in complemented form. The read operation is indicated by 1 on read/ write signal. Therefore during the read operation only the 'AND' gate 'c' becomes active. If the cell has been selected, then the output will become equal to the state of flip flop i.e. the data value stored in flip flop is read. In write operation 'a' & 'b' gates become active and they set or clear the J-K flip flop depending upon the input value. Please note in case input is 0, the flip flop will go to clear state and if input is 1, the flip flop will go to set state. In effect, the input data is

reflected in the state of flip-flop. Thus, we say that input data has been stored in flip-flop or binary cell.

Fig 4.22 is the extension of this binary cell to an IC RAM circuit, where a 2×4 decoder is used to select one of the four words. (For 4 words we need 2 address lines) Please note that each decoder output is connected to a 4bit word and the read/write signal is given to each binary cell. Once the decoder selects the word, the read/write input tells the operation. This is derived using an OR gate, since all the non-selected cells will produce a zero output. When the memory select input to decoder is 0, none of the words is selected and the contents of the cell are unchanged irrespective of read/write input.

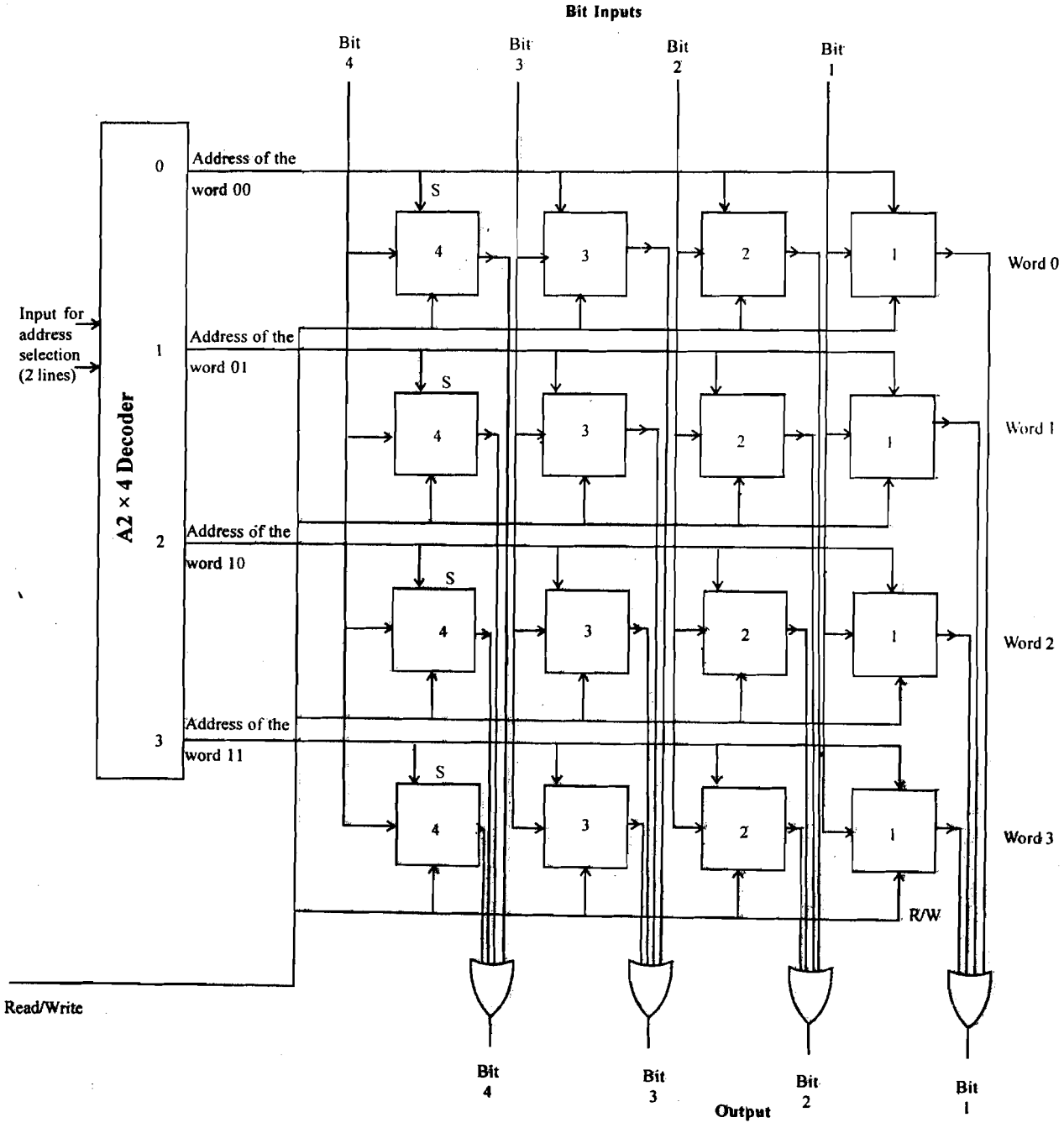


Figure 4.22: 4 × 4 RAM

After discussing so much about combinational circuits and sequential circuits, let us discuss in the next section an example having a combination of both circuits.

4.6 DESIGN OF A SAMPLE COUNTER

Let us design a synchronous BCD counter. A BCD counter follows a sequence of ten states and returns to 0 after the count of 9. These counters are also called **decade counters**. This type of counter is useful in display applications in which BCD is required for conversion to a decimal readout. Fig 4.23 shows the characteristic table for this counter.

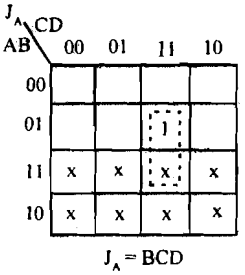
Present State				**Next State**				**Flip-Flops Inputs**							
A	B	C	D	A	B	C	D	J_A	K_A	J_B	K_B	J_C	K_C	J_D	K_D
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	0	X	0	X	0	X	1	X
1	0	0	1	0	0	0	0	0	X	0	X	0	X	X	1

Figure 4.23: Characteristic table for decade counter

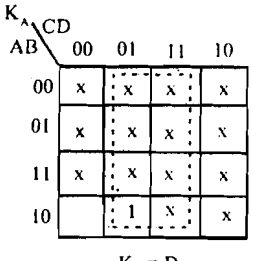
[*NOTE* : Remember excitation table for J-K flip flop given in fig 4.8]

There are 4 flip-flop inputs for decade counter i.e. A, B, C, D. The next state of flip-flop is given in the table. J_A & K_A indicates the flip flop input corresponding to flip-flop-A. Please note this counter require 4-flip-flops.

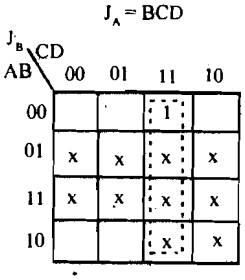
From this the flip flop input equations are simplified using K-Maps as shown in figure 4.24. The unused minterms from 1010 through 1111 are taken as don't care conditions.



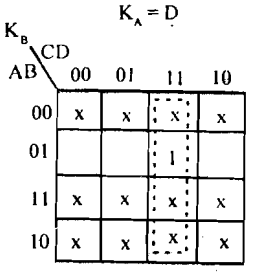
$J_A = BCD$



$K_A = D$



$J_B = CD$



$K_B = CD$

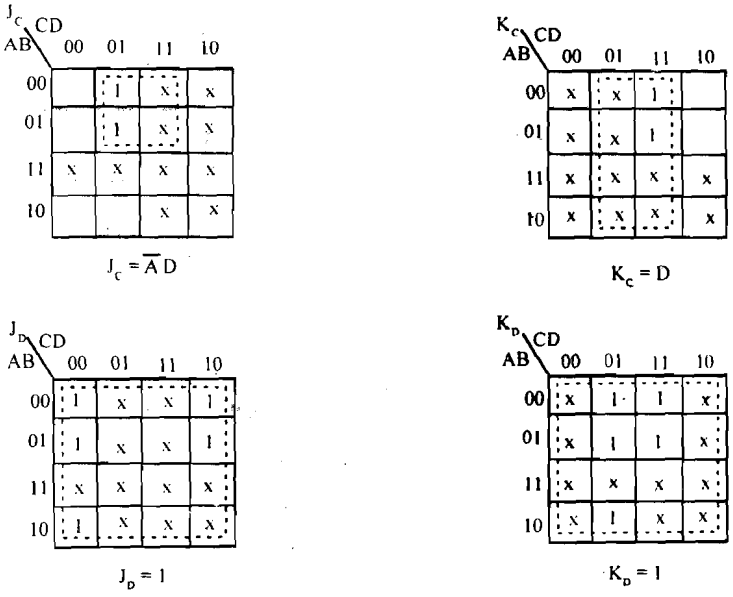


Figure 4.24: K-maps for Decade counter

Thus, the simplified input equation for BCD counter are:

J_A	=	BCD	K_A	=	D
J_B	=	CD	K_B	=	CD
J_C	=	$\overline{A}D$	K_C	=	D
J_D	=	1	K_D	=	1

The logic circuit can be made with 4 JK flip flops & 3 AND gates

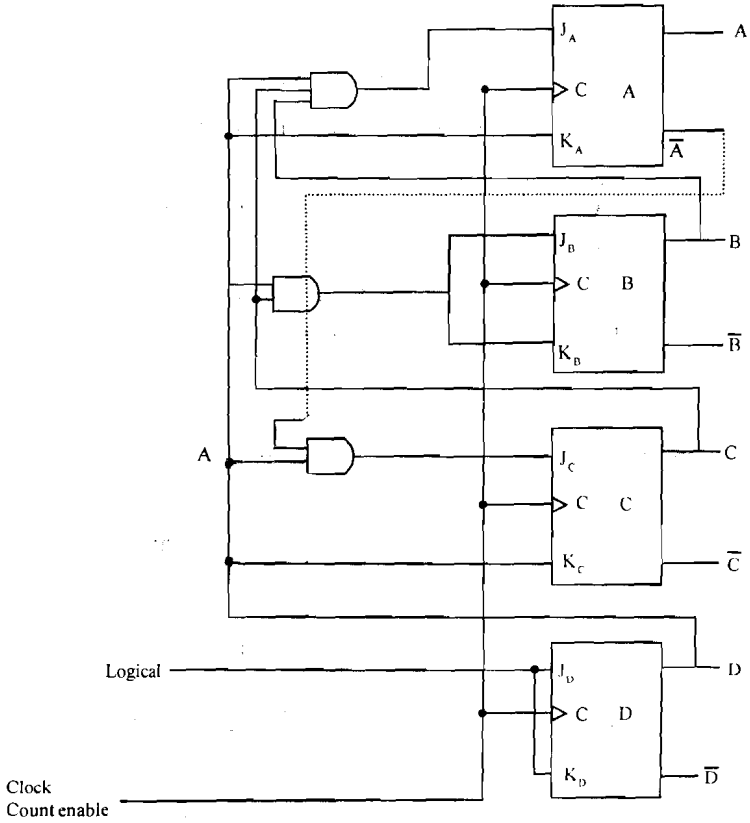


Figure 4.25: Logic Diagram for decade counter.

Check Your Progress 3

1) Differentiate between synchronous & asynchronous counters?

.....

.....

.....

2) Can ripple counter be constructed from a shift register?

.....

.....

.....

3) Can we design a counter with the following repeated binary sequence: 0,1,2,3,4,5,6. If yes, design it using J K flip flop.

.....

.....

.....

4.7 SUMMARY

As told to you earlier this unit provides you information regarding sequential circuits which is the foundation of digital design. Flip-flops are basic storage unit in sequential circuits are derived from the latches. The sequential circuit can be formed using combinational circuits (discussed in the last unit) and flip flops. The behavior of sequential circuit can be analyzed using tables & state diagrams.

Registers, counters etc. are structured sequential blocks. This unit has outlined the construction of registers, counters, RAM etc. Lastly, we discussed how a circuit can be designed using both sequential & combinational circuits. For more details, the students can refer to further reading.

4.8 SOLUTIONS / ANSWERS

Check Your Progress 1

1) An interconnection of combinational circuits and flip-flops, used for making different logic devices in computers that involves manipulation and storage of data. Some such circuits are registers, counters etc.

2) Flip flop is the basic storage element for synchronous sequential circuits. Whereas latches are bistable devices whose state normally depends upon the asynchronous inputs and are not suitable for use in synchronous sequential circuits using single clock

3) Excitation table indicates that if present and next state are known then what will be inputs whereas a characteristics table indicates just opposite of this i.e. inputs are known the, next state has to be found.

Check Your Progress 2

1) The main advantage is that they allow feed back paths

2) Edge-Triggered flip-flops are bi-stable devices with synchronous inputs whose state depends on the inputs only at the triggering transition of a clock pulse i.e. changes in output occur only at triggering transition of the clock

Check Your Progress 3

1) The main difference is the time when the counter flip-flops change its states. In synchronous counter all the flip flops that need to change; change simultaneously. In asynchronous counter the complement if to be done may ripple through a series of flip-flops.

2) Yes, but this: circuit will generate sequence of states where only 1-bit changes at a time i.e. 0000, 1000, 1100, 1110, 1111, 0111, 0011, 0001

3) Yes, We require 2^3 i.e. three flip flops for the sequence 0, 1, 2, 3, 4, 5&6.

Present State			Next State			Flip – Flops Inputs					
A	B	C	A	B	C	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	0	0	0	X	1	X	1	0	X

The state is don't care condition: Make the suitable K-maps. The following are the flip-flop input values:

$J_A = BC$ $\quad K_A = B$

$J_B = C$ $\quad K_B = C + A$

$J_C = \overline{A} + \overline{B}$ $\quad K_C = 1$

The circuit can be constructed as: