UNIT 2: DIGITAL COMPONENTS

UNIT STRUCTURE

2.1 Learning Objectives

2.2 Introduction

2.3 Combinational Circuits

- 2.3.1 Half-Adder
- 2.3.2 Full-Adder
- 2.3.3 Half-Subtractor
- 2.3.4 Full-Subtractor
- 2.3.5 Multiplexer
- 2.3.6 Demultiplexer
- 2.3.7 Encoder
- 2.3.8 Decoder
- 2.3.9 Magnitude Comparator

2.4 Sequential Circuits

2.5 Flip-Flops

- 2.5.1 RS Flip-Flop
- 2.5.2 D Flip-Flop
- 2.5.3 JK Flip-Flop
- 2.5.4 MS Flip-Flop

2.6 Counters

- 2.6.1 Asynchronous Counter
- 2.6.2 Synchronous Counter

2.7 Register

- 2.7.1 Serial In – Serial Out Register
- 2.7.2 Serial In – Parallel Out Register
- 2.7.3 Parallel In- Serial Out Register
- 2.7.4 Parallel In- Parallel Out Register

2.8 Let Us Sum Up

2.9 Answers to Check Your Progress

2.10 Further Readings

2.11 Model Questions

2.1 LEARNING OBJECTIVES

After going through this unit you will be able to:

- define combinational circuit and sequential circuit
- describe the working principle of Half-adder, Full-adder
- describe the working principle of Half-subtractor, Full-subtractor
- describe the working principle of Multiplexer, Demultiplexer
- describe the working principle of Encoder, Decoder
- describe the working principle of Flip-Flop, Register, Counter

2.2 INTRODUCTION

In the previous unit, we have discussed about the basic organization of the computer. In this unit, we will discuss the various combinational and sequential circuits to build your understanding on different digital components that are used in digital applications. A sound knowledge on these components will make you feel confident to understand the working principle of many digital devices in general and the computer system in particular. Combinational circuits, sequential circuits along with counters and registers are covered in detail in this unit. In this next unit, we will explore different data representation systems.

2.3 COMBINATIONAL CIRCUITS

To obtain a desired output according to a Boolean equation, various logic gates are often interconnected. These circuits are called combinational circuits. These are the combinations of different fundamental logic gates and hence the name Combinational Circuit. Its main characteristic is that the output is solely determined by the present inputs.

Inputs to a combinational circuit are given as either 0 or 1 and outputs are also available as either 0 or 1. In the process of designing a combinational circuit, first, a truth table is formed for the Boolean expression, that describes the combinational circuit. Next, the function is simplified and then the simplified function is implemented with gates to obtain the logic diagram of the combinational circuit. We may also obtain the Boolean expression from this logic diagram after determining the truth table.

2.3.1 Half-Adder

Half-adder is a circuit that can add two binary bits. Its outputs are SUM and CARRY. The following truth table shows the various combinations of inputs and the corresponding outputs of a half-adder. X & Y denote inputs and C & S denotes the two outputs CARRY & SUM.

Table 2.1: Truth Table for a Half-Adder

X	Y	CARRY (C)	SUM (S)
0	0	0	0
0	1	0	1 ($\overline{X}Y$)
1	0	0	1($X\overline{Y}$)
1	1	1 (XY)	0

The minterms for Sum and CARRY are shown in the bracket. The Sum-Of-Product equation for SUM is :

$$S = \overline{X}Y + X\overline{Y} \quad \ldots\ldots\ldots\ldots (1)$$
$$= X \oplus Y$$

Similarly, the SOP equation for the CARRY is:

$$C = XY \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

Combining the logic circuits for equation (1) & (2) we get the circuit for Half-Adder as :



Figure 2.1: Half-Adder Circuit and Symbol

2.3.2 Full-Adder

Full-Adder is a logic circuit to add three binary bits. Its outputs are SUM and CARRY. In the following truth table X,Y,Z are inputs and Ć and Ś are CARRY & SUM.

Table 2.2: Truth Table for Full- Adder

X	Y	Z	CARRY (Ć)	SUM (Ś)
0	0	0	0	0
0	0	1	0	1 ($\overline{X}\,\overline{Y}$Z)
0	1	0	0	1 ($\overline{X}$Y$\overline{Z}$)
0	1	1	1 ($\overline{X}$YZ)	0
1	0	0	0	1 (X$\overline{Y}\,\overline{Z}$)
1	0	1	1 (X$\overline{Y}$Z)	0
1	1	0	1 (XY$\overline{Z}$)	0
1	1	1	1 (XYZ)	1 (XYZ)

The minterms are written in the brackets for each 1 output in the truth table. From these, the SOP equation for full summation can be written as:

$$\acute{S} = \overline{X}\,\overline{Y}Z + \overline{X}Y\overline{Z} + X\overline{Y}\,\overline{Z} + XYZ$$
$$= \overline{X}\,(\overline{Y}Z + Y\overline{Z}) + X(\overline{Y}\,\overline{Z} + YZ)$$
$$= \overline{X}S + X\overline{S} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

Here, S is SUM of Half-Adder.

Again, SOP equation for Full–Adder CARRY is :

$$\acute{C} = \overline{X}YZ + X\overline{Y}Z + XY\overline{Z} + XYZ$$
$$= \overline{X}YZ + XYZ + X\overline{Y}Z + XY\overline{Z}$$
$$= (\overline{X} + X)YZ + X(\overline{Y}Z + Y\overline{Z})$$
$$= YZ + XS$$
$$= C + XS \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4)$$

Here also, C means CARRY of half-adder and S means SUM of half-adder.

Now, using two half-adder circuits and one OR gate we can implement equation (3) and (4) to obtain a full-adder circuit as follows.

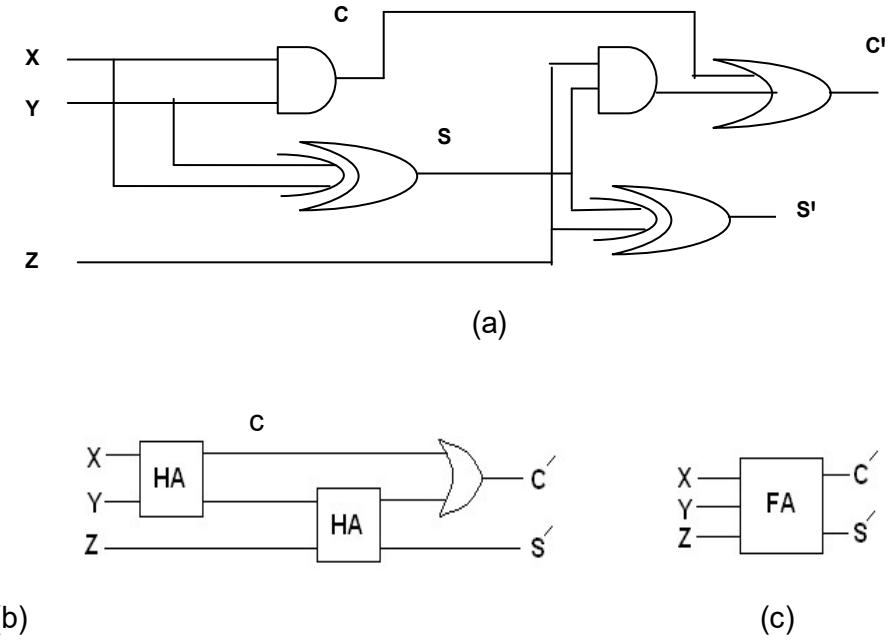


Figure 2.2: Full-Adder Circuit

(a) logic diagram, (b) block diagram (c) Symbol

2.3.3 Half - Subtractor

A half-subtractor subtracts one bit from another bit. It has two outputs, DIFFERENCE (D) and BORROW (B).

Table 2.3: Truth Table for Half-Subtractor

X	Y	BORROW(B)	DIFFERENCE (D)
0	0	0	0
0	1	1 ($\overline{X}$ Y)	1 ($\overline{X}$ Y)
1	0	0	1 (X $\overline{Y}$)
1	1	0	0

The mean terms are written within parenthesis for output 1 in each column. The SOP equations are :

$$D = \overline{X}Y + X\overline{Y} = X \oplus Y$$

$$= S \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (5)$$

$$B = \overline{X}Y \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (6)$$

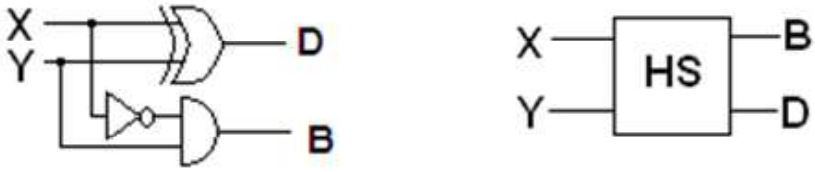


Figure 2.3: The Half-Subtractor circuit and symbol

2.3.4 Full-Subtractor

A full-subtractor circuit finds the Difference and Borrow on the subtraction operation involving three binary bits.

Table 2.4: Truth Table of Full-Subtractor

X	Y	Z	BORROW (B´)	DIFFERENCE(D´)
0	0	0	0	0
0	0	1	1 ($\overline{X}\,\overline{Y}\,Z$)	1 ($\overline{Y}\,\overline{X}\,Z$)
0	1	0	1 ($\overline{X}\,Y\,\overline{Z}$)	1 ($\overline{X}\,Y\,\overline{Z}$)
0	1	1	1 ($\overline{X}\,YZ$)	0
1	0	0	0	1 ($X\,\overline{Y}\,\overline{Z}$)
1	0	1	0	0
1	1	0	0	0
1	1	1	1 (XYZ)	1 (XYZ)

The SOP equation for the DIFFERENCE is :

$$D' = \overline{X}\,\overline{Y}\,Z + \overline{X}\,Y\,\overline{Z} + X\,\overline{Y}\,\overline{Z} + XYZ$$

$$= \overline{X}\,Y\,\overline{Z} + X\,\overline{Y}\,\overline{Z} + XYZ + \overline{X}\,\overline{Y}\,Z$$

$$= (\overline{X}\,Y + X\,\overline{Y})\,\overline{Z} + (XY + \overline{X}\,\overline{Y})\,Z$$

$$= D\,\overline{Z} + \overline{D}\,Z \quad \ldots\ldots\ldots\ldots\ldots\ldots (7)$$

And SOP equation for BORROW is :

$$B' = \overline{X}\,\overline{Y}\,Z + \overline{X}\,Y\,\overline{Z} + \overline{X}\,YZ + XYZ$$

$$= \overline{X}\,\overline{Y}\,Z + XYZ + \overline{X}\,Y\,\overline{Z} + \overline{X}\,YZ$$

$$= (\overline{X}\,\overline{Y} + XY)\,Z + \overline{X}\,Y\,(\overline{Z} + Z)$$

$$= \overline{D}\,Z + \overline{X}\,Y \quad \ldots\ldots\ldots\ldots\ldots\ldots (8)$$

In equation (7) and (8) , D stands for DIFFERENCE of half-subtractor. Now, from the equations (7) and (8) we can construct a full-subtractor using two half-subtractor and an OR gate.

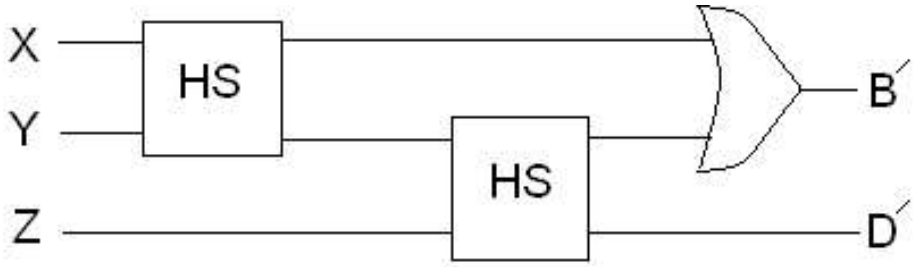
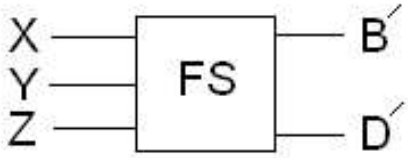


Figure 2.4: Full-Subtractor Circuit

The symbol of full-subtractor is :





CHECK YOUR PROGRESS

Q1. A half-adder can add

(a) Two binary number of 4 bit each (b) Two binary bit

(c) Add half of a binary number (d) None of these

Q2. A full-adder is a logic circuit that has two outputs namely:

(a) product & sum (b) sum & borrow

(c) sum & carry (d) carry & borrow

Q3. A half-subtractor can perform

(a) subtraction of two binary bits (b) product of two binary bits

(c) complement of half binary bits (d) none of these

Q4. A full-subtractor has the ability to do

(a) subtraction of two binary numbers (b) subtraction of three binary bits

(c) product of three binary bits (d) division three binary bits

2.3.5 Multiplexer

Multiplexer is a circuit which has many inputs and only one output. Multiplexer can select any one of its many inputs by applying a control signal and steer the selected input to the output. Generalized block diagram of a multiplexer is shown in figure 2.5.

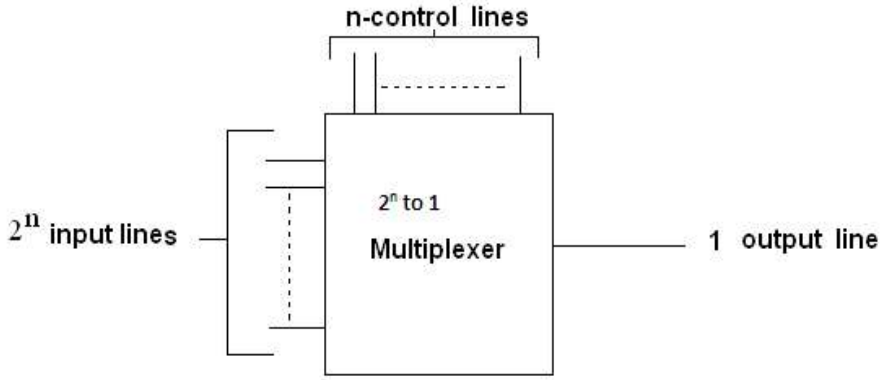


Figure 2.5: Block Diagram of Multiplexer.

It has 2^n inputs, n – numbers of control or selection lines and only one output. A multiplexer is also called a many – to – one data selector.

8-to-1 MULTIPLEXER:

Figure 2.6 shows a 8-to-1 multiplexer, where there are 8 inputs, 3 control or selection lines and 1 output. The eight inputs are labeled as $I_0, I_1, I_2, I_3, I_4, I_5, I_6, I_7$ and the selection lines are labeled as A, B, C, where input is steered to the output and depends on the value of ABC. For example, if ABC = 000, then the upper AND gate is enabled and all other AND gates are disabled. As a result, the input I_0 alone is steered to the output. Similarly if ABC = 110 then the AND gate connected to the data line I_6 is enabled while all the other AND gates are disabled. Therefore, the input I_6 appears at the output. Hence when ABC = 110 , the output is Y = I_6 .

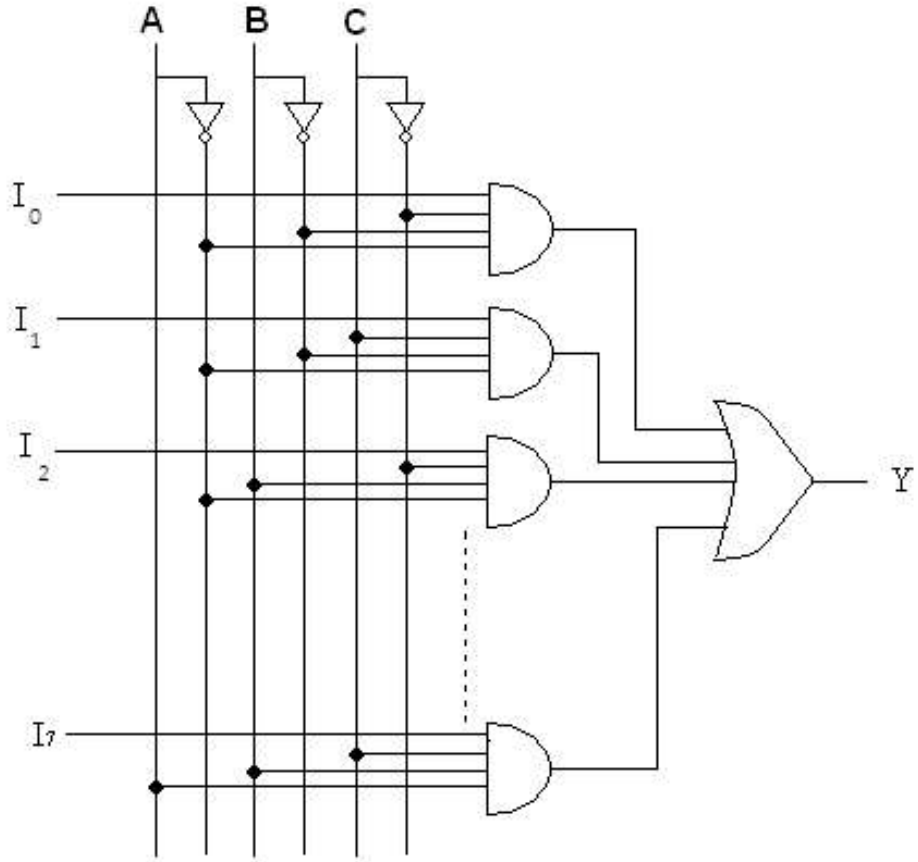


Figure 2.6: 8-to-1 Multiplexer

16-to-1 MULTIPLEXER

Figure 2.7 shows a 16 – to – 1 multiplexer. In this circuit, there are 16 data input lines, 4 control lines and 1 output. The input lines are denoted by $I_0, I_1, I_2, I_3, \ldots\ldots\ldots\ldots I_{15}$ and the control lines are donated by ABCD. The output is denoted by Y.

Out of 16 input lines, only one is transmitted to the output depending upon the value of ABCD. If ABCD = 0000 then I_0 is steered to the output since the upper AND gate is enabled alone and all others are disabled. Similarly if ABCD = 0010 then I_2 appears at the output. If ABCD = 1111 then the last AND gate is only enabled and therefore I_{15} appears at the output.

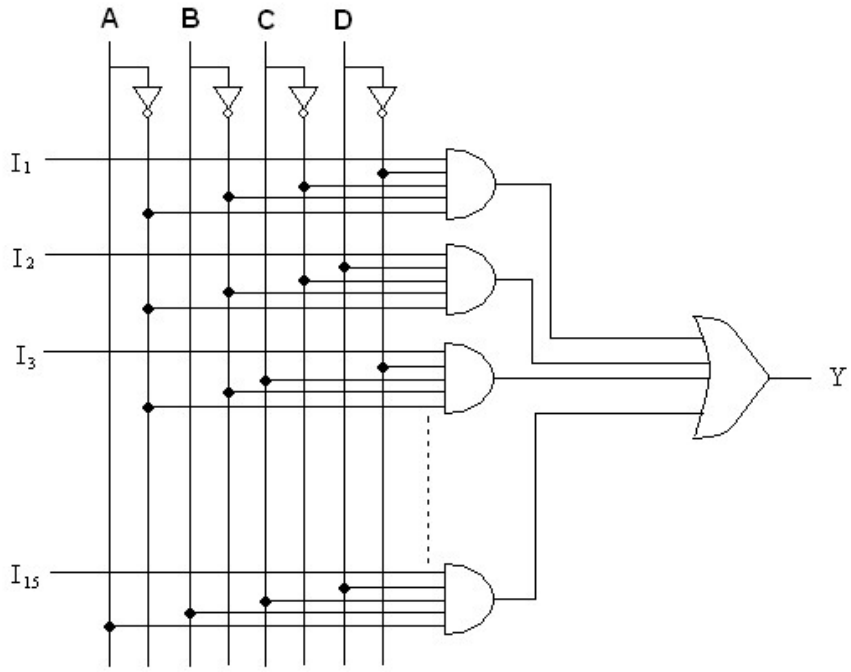


Figure 2.7: 16-to-1Multiplexer

2.3.6 De-Multiplexer

De-multiplexer means One-to-Many. It is opposite to the multiplexer. It has 1 input and many outputs. With the application of appropriate control signal, the common input data can be steered to one of the output lines. Figure 2.8 shows a generalized block diagram of a de-multiplexer.

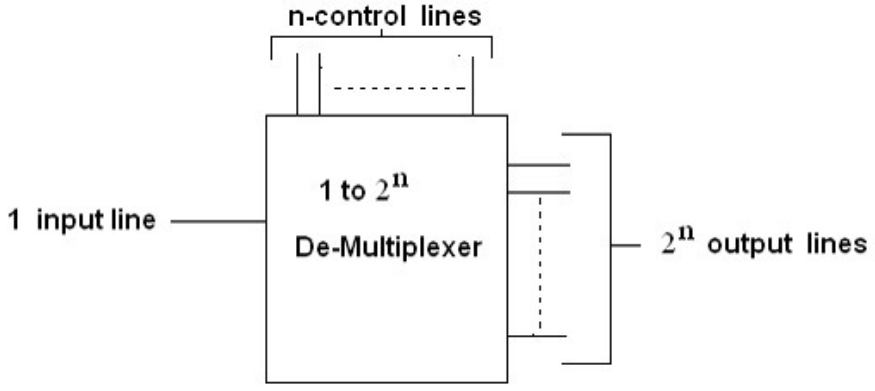


Figure 2.8: Block diagram of a De-Multiplexer

To have 2^n output lines, there must be n control lines in a de-multiplexer.

1-to-16 DE-MULTIPLEXER

In figure 2.9 we have shown a 1-to- 16 de-multiplexer.

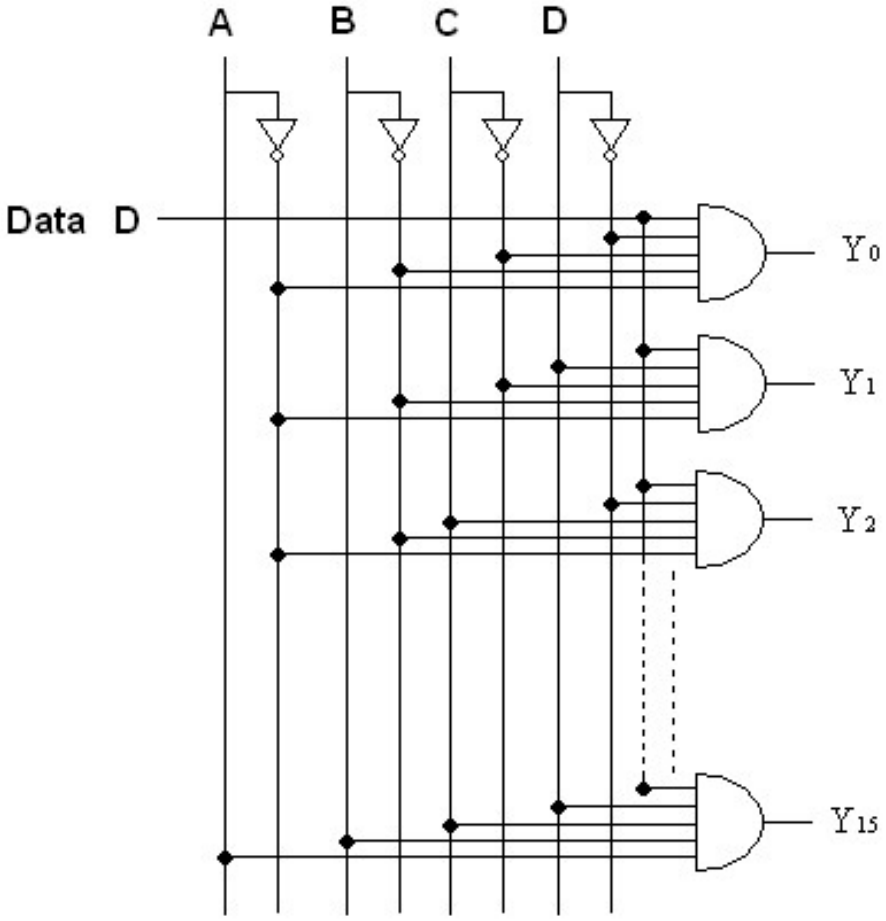


Figure 2.9: 1-to-16 De-Multiplexer

Here the data input line is denoted by D. This input line is connected to all the AND gates through which output appears. Depending upon the control signal, only one AND gate becomes enabled and the data input D appears through that AND gate. So, when ABCD = 0000, the upper AND gate is enabled and data input D appears at Y_0 as output.

When ABCD = 1111, the bottom AND gate becomes enabled and D appears at Y_{15} as output. For other combinations, D' appears at other output terminal.



CHECK YOUR PROGRESS

Q5. Multiplexer means _____

(a) multiple to many (b) one to many

(c) many to one (d) one to one

Q6. A 16-to-1 multiplexer has _____

(a) 1 control lines (b) 2 control lines

(c) 3 control lines (d) 4 control lines

Q7. De-multiplexer means _____

(a) deduct multiple bits (b) one-to-many

(c) multiple-to-multiple (d) one-to-one

2.3.7 Encoder

An encoder converts a digital signal into a coded signal. A generalized view of an encoder is shown in figure 2.10

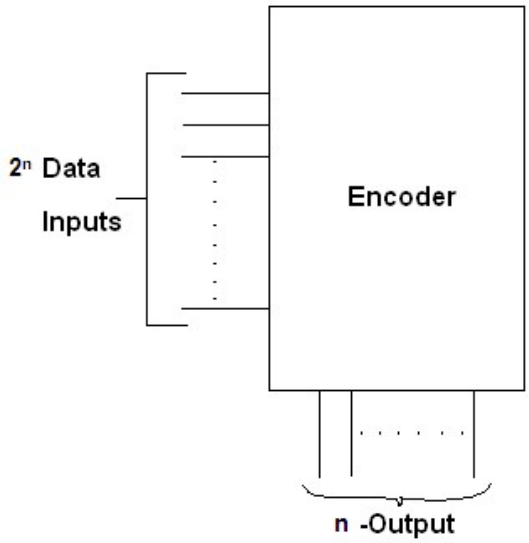


Figure 2.10: Block Diagram of an Encoder

In figure 2.10, we can see that there are 2^n input lines and n – numbers of output lines. Out of 2nd inputs, only one input line is active at a time. Encoder generates a coded output which is unique for each of the active input.

DECIMAL- TO -BCD ENCODER

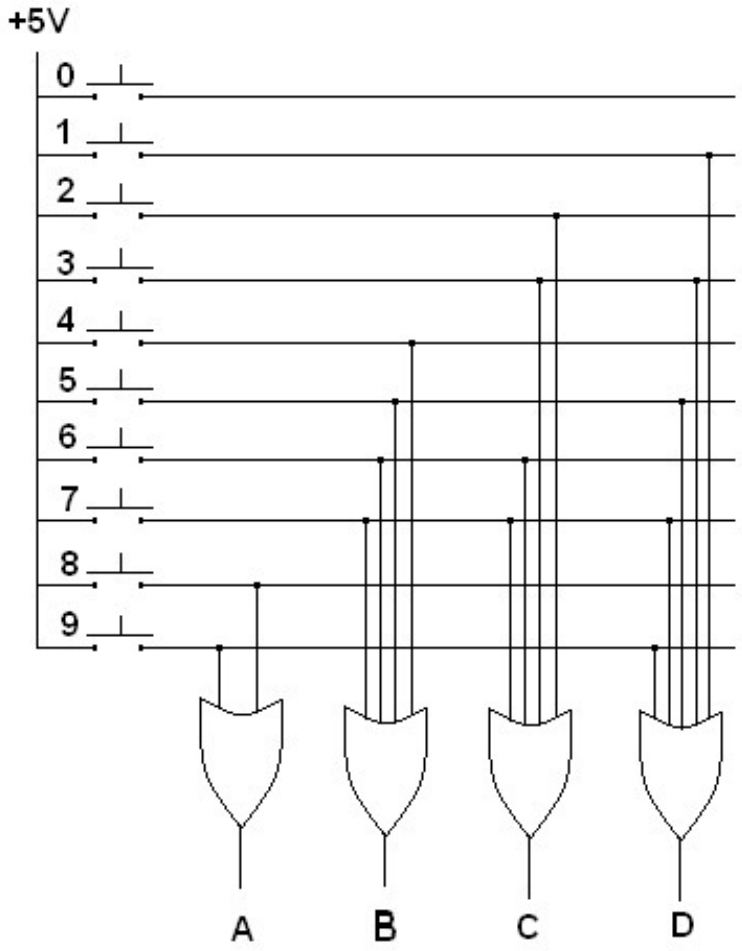


Figure 2.11: Decimal-to-BCD Encoder

A decimal to BCD encoder is shown in Fig 2.11 This circuit generates BCD output when any one of the push button switches is pressed. For example, if button 6 is pressed, the B and C OR gates have high inputs and the corresponding output becomes

ABCD = 0110

If button 8 is pressed, the OR gate A receives a high input and therefore the output becomes

ABCD = 1000

2.3.8 Decoder

A decoder is a digital circuit which has n-input lines and 2^n output lines. A decoder and a de-multiplexer has similarity. In a de-multiplexer, there is a single input line connected to every output 'AND' gate whereas in a decoder the input line is absent.

A 3–to–8 decoder as shown in Figure 2.12. Its truth table is shown below:

Table 2.5: Truth table for a 3-to-8 decoder

Inputs			Outputs							
A	**B**	**C**	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
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

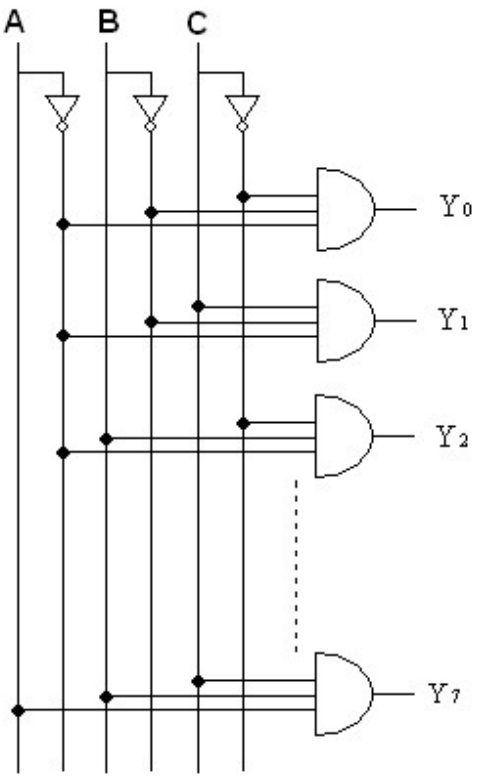


Figure 2.12: 3-to-8 decoder

2.3.9 Magnitude Comparator

A magnitude comparator circuit can compare two binary numbers to determine which one is greater than the other or their equality. Such a magnitude comparator has three output lines for A > B, A = B, A < B where A & B are two n-bits binary numbers. Every bit of one number is compared with the corresponding bit of the other number by ExOR gate.

A 4-bit magnitude comparator, SN 7485 is available in chip form. The block diagram of SN 7485 is shown in figure 2.13. It compares two 4-bit binary numbers $A_3 A_2 A_1 A_0$ and $B_3 B_2 B_1 B_0$. Three output terminals are available for A < B, A = B and A > B.

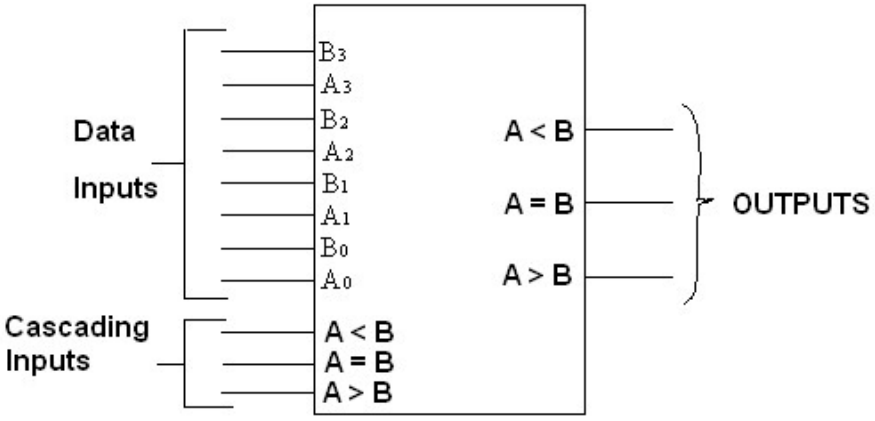


Figure 2.13: Two 4-bit words magnitude comparator SN 7485

Table 2.6: Function Table SN 7485

Comparing Inputs				Cascading Inputs			Outputs		
A_3B_3	A_2B_2	A_1B_1	A_0B_0	**A > B**	**A < B**	**A = B**	**A > B**	**A < B**	**A = B**
$A_3 > B_3$	X	X	X	X	X	X	H	L	L
$A_3 < B_3$	X	X	X	X	X	X	L	H	L
$A_3 = B_3$	$A_2 > B_2$	X	X	X	X	X	H	L	L
$A_3 = B_3$	$A_2 < B_2$	X	X	X	X	X	L	H	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 > B_1$	X	X	X	X	H	L	L
$A_3 = B_3$		$A_1 < B_1$	X	X	X	X	L	H	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 > B_0$	X	X	X	H	L	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 < B_0$	X	X	X	L	H	L
$A_3 = B_3$		$A_1 = B_1$	$A_0 = B_0$	H	L	L	H	L	L
$A_3 = B_3$		$A_1 = B_1$	$A_0 = B_0$	L	H	L	L	H	L
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	L	L	H	L	L	H
	$A_2 = B_2$								
	$A_2 = B_2$								
	$A_2 = B_2$								

To compare any number having more than 4-bits, two or more such chip can be cascaded. The A > B, A < B and A = B outputs of a stage that handles less significant bits are connected to the corresponding cascading inputs of the next stage that handles the more significant bits.



CHECK YOUR PROGRESS

Q8. An encoder–

(a) converts a digital input to another form of digital output.

(b) converts analog input to digital output

(c) selects one out of many inputs.

(d) none of these.

Q9. Decoder has n inputs line and–

(a) n output lines.

(b) 2^n output lines.

(c) n^2 output lines.

(d) no output lines.

Q10. Magnitude comparator–

(a) compares two multi bit binary number.

(b) magnify any digital signal.

(c) compress binary numbers.

(d) check error in a binary number.

2.4 SEQUENTIAL CIRCUITS

If the output of a circuit depends on its present inputs and the immediate past output, then the circuit is called **sequential circuit**. To build a sequential circuit, we need memory circuits and combinational circuits. Flip-Flop is used as memory circuit, the application of which we would see in counter, register etc.

2.5 FLIP-FLOPS

A digital circuit that can produce two states of output, either high or low is called a multivibrator. There are three types of multivibrators. They are monostable, bi-stable and a stable.

A Flip-Flop is a bi-stable multivibrator and therefore it has two stable states of output-either high or low. Depending on its inputs and previous output, its new output is either high (or 1) or low (or 0). Once the output is fixed, the inputs can be removed and then also the already fixed output will be retained by the flip-flop. Hence, a flip-flop can be used as basic circuit of memory for storing for one bit of data. To store multiple bits we can use multiple numbers of Flip-Flop. These are also used to build counter, register etc.

There are many types of Flip-Flops. Some of them are:

- RS Flip-flop
- D Flip-Flop
- JK Flip-Flop
- MS Flip-Flop

2.5.1 RS Flip-Flop

RS Flip-Flop is also called Set-Reset Flip-Flop. A RS flip-flop can be built using many different circuits. Here, we have shown two RS flip-flop circuits using NOR and NAND gate. To study its working principle, we can use any one of them. In the following section, let us consider the RS flip-flop using NOR gate.

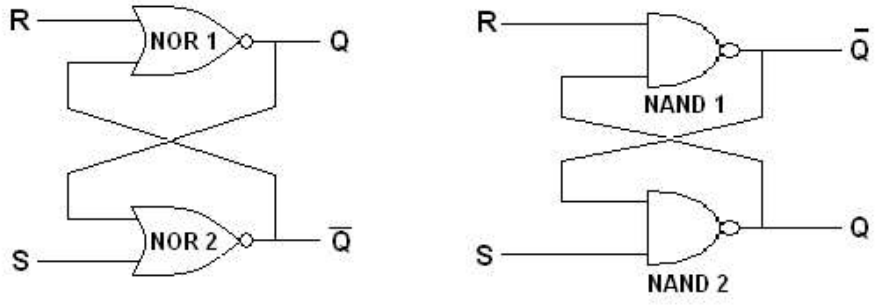


Figure 2.14: Basic circuit of RS Flip-Flop using NOR and NAND gate

RS flip-flop has two inputs —Set(s) and Reset(R). It has two outputs, Q and $\overline{Q}$. It should be noted that $\overline{Q}$ is always the complement of Q. Various combinations of inputs and their corresponding outputs are listed in the truth table below:

Table 2.7: Truth Table of RS Flip-Flop with NOR gates

R	S	Q	Action
0	0	Last value	No change
0	1	1	Set
1	0	0	Reset
1	1	?	Forbidden

The first input condition in the table is R=0, S=0. Since a 0 input has no effect on its output, the Flip-Flop retains its previous state. Hence Q remains unchanged.

The second input condition R=0, S=1 forces the output of NOR gate2 low. This low output will reach NOR gate1 and when both inputs of NOR gates1 are low, its inputs Q will be high. Thus a 1 at the S input will SET the flip-flop and Q will be equal to 1.

The third input condition R=1, S=0 will force the output of NOR gate1 to low. This low will reach NOR gate2 and force its outputs to high. Hence, when R=1, S=0, then Q=0, $\overline{Q}$=1. Thus the flip-flop is RESET.

The last input condition in the table R=1, S=1 is forbidden since it forces both the NOR gates to the low state. It means both Q=0, and =0 at the same time, which violates the basic definition of flip-flop that requires Q to be the complement of Q Hence this input condition is forbidden and its output is unpredictable.

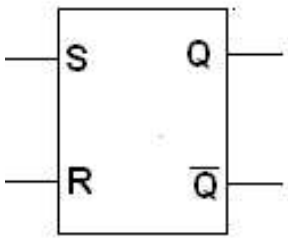


Figure 2.15: Symbol of RS Flip-Flop

CLOCK INPUT

For synchronization of operation of multiple flip-flop, an additional signal is added to all types of flip-flop. It is called clock signal, generally abbreviated as CLK. Addition of CLK signal ensures that whatever may be the input to the flip-flop, it affects the output only when CLK signal is given. Figure 2.16 shows a clocked RS flip-flop.

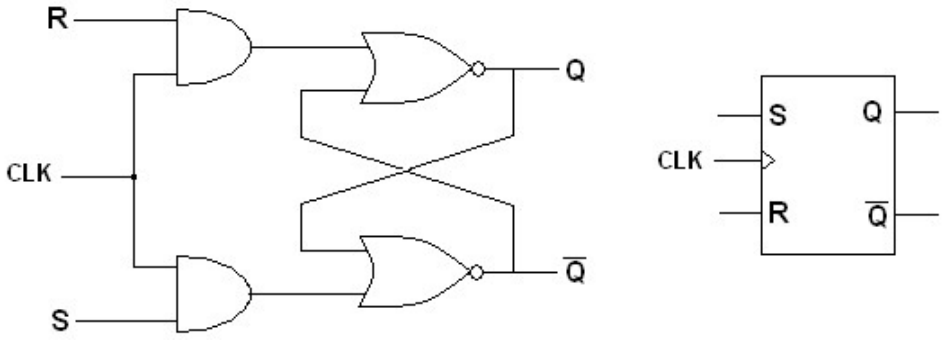


Figure 2.16: Basic circuit and symbol of Clocked RS Flip-Flop

2.5.2 D Flip-Flop

D flip-flop is a modification of RS flip-flop. In RS flip-flop, when both the inputs are high, R=1, S=1 and the output becomes unpredictable and this input combination is termed as forbidden. To avoid this situation, the RS flip-flop is modified so that both the inputs cannot be same at the same time. The modified flip-flop is called D flip-flop. Figure 2.17 shows a clocked D flip-flop.

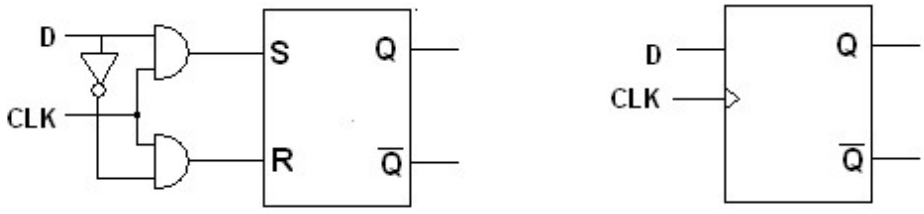


Figure 2.17: (a) Clocked D Flip-Flop (b) Symbol of D Flip-Flop

In D flip-flop both inputs of RS flip-flip are combined together to make it one by a NOT gate so that inputs can not be same at a time. Hence, in D flip-flop there is only one input. The truth table is given in table 2.8.

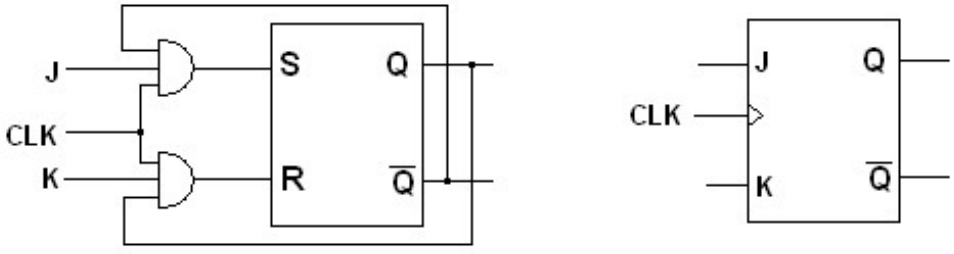
Table 2.8: Truth Table of D Flip-Flop

CLK	D	Q
0	X	Last state
1	0	0
1	1	1

In a clocked D flip-flop the value of D cannot reach the output Q when the clock pulse is low. During a low clock, both AND gates are disabled, therefore, D can change value without affecting the value of Q and this fact is expressed by putting an “X” as D input in the truth table 2.8. On the other hand, when the clock is high, both AND gates are enabled. In this situation, Q is forced to be equal to the value of D. In another way, we can say that in the D flip-flop above, Q follows the value of D while the clock is high. This kind of D flip-flop is often called a D latch.

2.5.3 JK Flip-Flop

In RS flip-flop, the input R= S=1 is called forbidden as it causes an unpredictable output. In JK flip-flop this condition is used by changing the RS flip-flop in some way. In JK flip-flop both input can be high simultaneously and the corresponding toggle output makes the JK flip-flop a good choice to build counter- a circuit that counts the number of +ve or –ve clock edges. Figure 2.18 shows one way to build a JK flip-flop.



(a) (b)

Figure 2.18: (a) JK Flip-Flop (b) Symbol of JK Flip-Flop

Table 2.9: Truth table for JK Flip-Flop

CLK	J	K	Q
X	0	0	Last state
1	0	1	0
1	1	0	1
1	1	1	Toggle

The inputs J and K are called control inputs because their combinations decide what will be the output of JK flip-flop when a +ve clock pulse arrives. When J and K are both low, both the AND gates are disabled. Therefore the CLK pulse has no effect. The first input combination of the truth table shows this and under this case the output Q retains its last state i.e. no change of state.

When J is low, K is high, the upper AND gate is disabled while the lower AND gate is enabled. Hence the flip-flop cannot be set; instead it is reset, i.e. Q=0. This is shown by the second entry in the truth table.

When J is high, K is low the upper AND gate is enabled while the lower one is disabled. So the flip-flop is set there by making Q=1.

When J and K are both high, then the flip-flop is set or reset depending on the previous value of Q. If Q is high previously, the lower AND gate sends a RESET trigger to the flip-flop on the next clock pulse. Then Q becomes equal to 0. On the other hand if Q is low previously, the upper AND gate sends a SET trigger on the flip-flop making Q=1.

So, when J = K = 1, Q changes its value from 0 to 1 or 1 to 0 on the positive clock pulse. This changing of Q to $\overline{Q}$ or to Q is called toggle. Toggle means to change to the opposite state.

Any flip-flop may be driven by +ve as well as –ve clock. As such JK flip-flop can also be driven by positive clock as well as negative clock. Figure 2.19 shows symbol of positive clocked and negative clocked JK flip-flop.

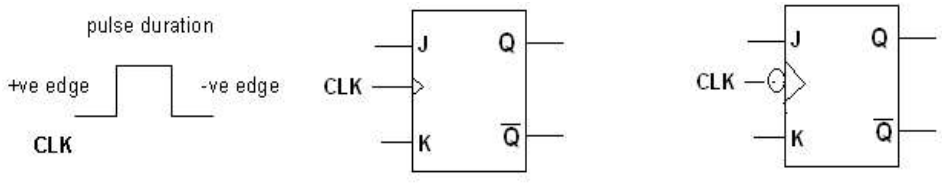


Figure 2.19: Positive and Negative clocked JK Flip-Flop

If a JK flip flop is driven by clock pulse duration, due to the feedback connection as shown in fig 2.18(a), the state of the flip flop i.e. value of Q, may be repeatedly change many times in a single clock pulse. So, it is highly desirable to make the flip flop sensitive to driven by pulse transition not pulse duration.

RC Differentiator Circuit:

The clock pulse applied to a flip-flop is a square wave signal. Clock pulse is used to achieve synchronization of flip-flop operation. To make the synchronization more precise, the square wave pulse is further modified to make it a narrow spike by using a RC differentiator circuit as shown in figure 2.20.

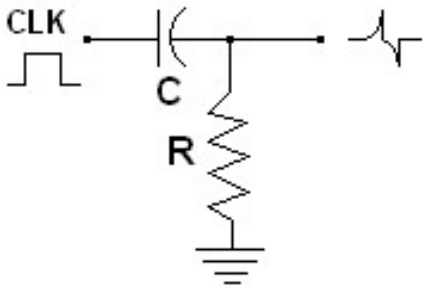


Figure 2.20 : RC Differentiator Circuit

The upper tip of the differentiated pulse is called positive edge and the lower tip is called negative edge. When a flip-flop is triggered by this type of narrow spike, it is called edge triggered flip-flop. If the flip-flop is driven by +ve edge, it is called +ve edge triggered flip-flop. If it is a driven by negative edge, it is called negative edge triggered flip-flop.

Racing

In a flip-flop if the output toggles more than once during a clock edge, then it is called racing. All flip-flop has a propagation delay, means the output changes its state after a certain time period from applying the input and the clock pulse. So, when a flip-flop is edge triggered, then due to propagation delay the output cannot affect the input again because, by that time, the edge of clock pulse has already passed away. If the propagation delay of a flip-flop is 20 ns and the width of the spike is less than 20 ns, then the returning Q and $\overline{Q}$ arrive too late to cause false triggering.

2.5.4 MS Flip-Flop

MS flip-flop is another way to avoid racing. Figure 2.21 shows one way to build MS flip-flop using two JK flip-flop, one of which is positive edge triggered and the other is negative edge triggered. The first JK flip-flop is master and the later is called slave. The master responds to its J and K inputs at the positive edge. J=1, K=0, the master sets on the positive clock edge. The high Q output of the master drive the j input of the slave. So, at the negative clock edge, the slave also sets, copying the action of the master.

When j=0, K=1, master resets at +ve clock edge and the slave resets of the –ve clock edge.

When j=K=1 master toggles at +ve clock edge, and the slave toggles at the –ve clock edge.

Hence whatever master does, the slave copies it.

MS flip-flop is a very popular flip-flop in industry due to its inherent resistance to racing. Hence it is extensively used to build counters.

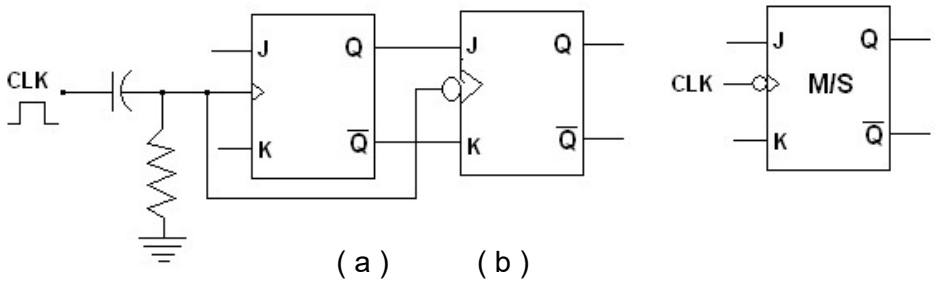


Figure 2.21: (a) Edge triggered JK MS Flip-Flop
(b) Symbol of JK MS Flip-Flop



CHECK YOUR PROGRESS

Q11. A flip-flop is basically a _____

(a) mono-stable multi-vibrator (b) a stable multi-vibrator

(c) bi-stable multi-vibrator (d) none of these

Q12. JK flip-flop has the specialty in _____

(a) fast response time (b) toggle property

(c) spike shaped clock input (d) preset input

Q13. In MS flip-flop the master changes state _____

(a) after the slave (b) with the slave at the same time

(c) before the slave (d) never

2.6 COUNTER

A counter is one of the most useful sequential circuits in a digital system. A counter driven by a clock can be used to count the number of clock cycles. Since the clock pulse has a definite time period, the counter can be used to measure time, the time period or frequency.

There are basically two types of counter— *Synchronous counter* and *asynchronous counter*.

Counters are constructed by using flip-flops and other logic gates. If the flip-flops are connected serially then the output of one flip-flop is applied as input to the next flip-flop. Therefore, this type of counter has a cumulative setting time due to propagation delay. Counter of this type are called serial or asynchronous counter. These counters have speed limitation.

Speed can be increased by using parallel or synchronous counter. Here, flip-flops are triggered by a clock at a time and thus the setting time is equal to the propagation delay of a single flip-flop. But this type of synchronous counters require more hardware and hence it is costly.

Combination of serial or parallel counter is also done to get an optimize solution of speed and hardware/cost. If each clock pulse advances the contents of the counter by one, it is called up counter. If the content of the counter goes down at each clock pulse, it is called down counter.

Before operation, some time it is required to reset all the flip-flops to zero. It is called “Clear”. Some time, it is required to set the flip-flops. It is called preset. To do these, two extra inputs are there in every flip-flop called CLR and PR.

2.6.1 Asynchronous Counter

When the output of a flip-flop is used as the clock input for the next flip-flops it is called asynchronous counter.

Asynchronous counters are also called ripple counter because flip-flop transitions ripple through from one flip-flop to the next in sequence until all flip-flop reach a new state.

A binary ripple counter can be constructed by using clocked JK flip-flop. Figure 2.22(a) shows three MS JK flip-flops connected in series. The clock drives flip-flop A. The output of A drives B and the output of B drives C. J and K inputs of all the flip-flops are connected to positive to make them equal to 1. Under this condition each flip-flop will change state (toggle) with a negative transition at its clock point.

In the counter shown in Figure 2.22 (a) the flip-flop A changes its state at the negative edges of the clock pulses. Its output is applied to the B flip-flop as its clock input.

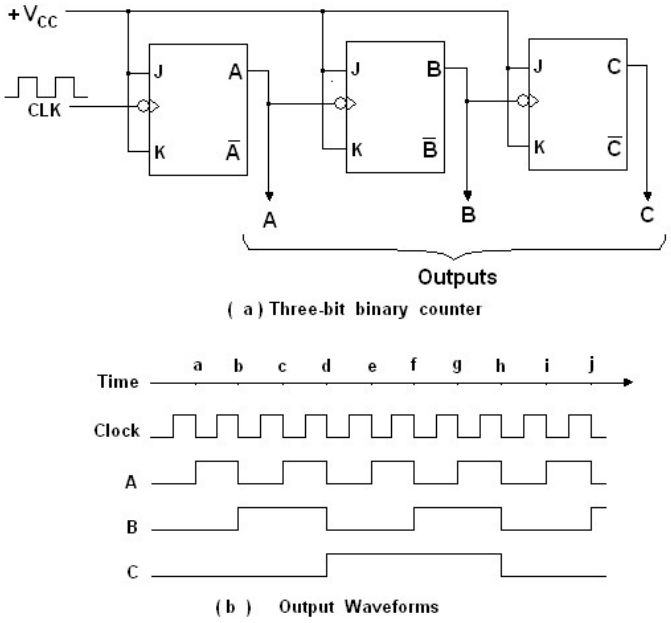


Figure 2.22: Three Bit Binary Counter

The output of B flip-flop toggles at the negative edges of the output of A flip-flop. Similarly output of B flip-flop is used as clock input of the C flip-flop and therefore C toggles at the negative edges of the output of B flip-flop. We can see that triggering pulses move through the flip-flops like a ripple in water.

The wave form of the ripple counter is shown in Figure 2.22(b). It shows the action of the counter as the clock runs. To understand the wave form, let us assume that the counter is cleared before the operation. The A output is assumed as the list significant bit(LSB) and C is the most-significant-bit (MSB). Hence, at very beginning the contents of the counter is CBA=000.

Flip-flop A changes its state to 1 after the negative pulse transition. Thus, at point a on the time line, A goes high. At point b, to it goes low, at c it goes back to high and so on.

Now, output of A acts as clock input of B. So, each time the output of A goes low, flip-flop B will toggle. Thus, at point b on the time line, B goes high, at point d it goes low, and toggles back high again at point f and so on.

Since B acts as the clock input for C, each time the output of B goes low, the C flip-flop toggles. Thus, C goes high at point d on the time line, it goes back to low again at point h.

We can see that the output wave form of A has half the frequency of the clock input wave. B has half the frequency than that of A and C has half the frequency than that of B.

We can further see that since the counter has 3 flip-flops cascaded together, it progresses through 000—— 001——010—-011—100—101—110—111 as its CBA output. After CBA= 111, it starts the cycle again from CBA=000. One cycle from 000—111 takes 8 clock pulses, as it is evident from the wave form as well as from the truth table.

2.6.2 Synchronous Counter

An asynchronous counter or ripple counter has limitation in its operating frequency. Each flip-flop has a delay time which is additive in asynchronous counter. In synchronous counter the delay of asynchronous counter is overcome by the use of simultaneous applications of clock pulse to all the flip-flops. Hence, in synchronous counter, the common clock pulse triggers all the flip-flops simultaneously and, therefore, the individual delay of flip-flop does not add together. This feature increases the speed of

synchronous counter. The clock pulse applied can be counted by the output of the counter.

To build a synchronous counter, flip-flops and some additional logic gates are required. Figure 2.23 shows a three stage synchronous or parallel binary counter along with its output wave forms and truth table. Here the J and K inputs of each flip-flop is kept high and, therefore, the flip-flops toggle at the negative clock transition at its clock input. From figure 2.23 we can see that the output of A is ANDed with CLK to drive the 2nd flip-flop and the outputs of A, B are ANDed with CLK to drive the third flip-flop. This logic configuration is often referred to as “steering logic” since the clock pulses are steered to each individual flip-flop.

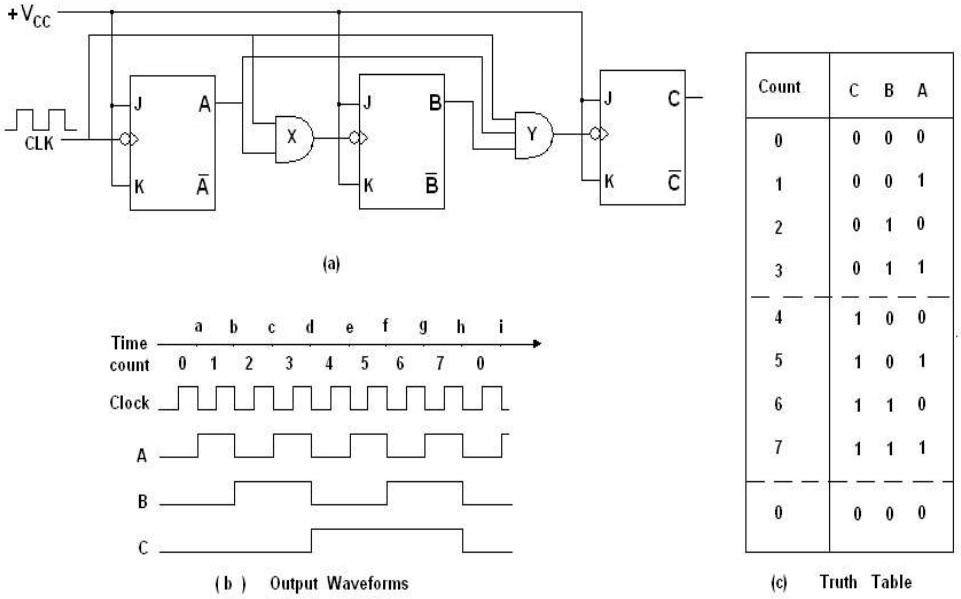


Figure 2.23: Parallel Binary Counter

The AND gate Y is enabled only when both A and B are high and it transmits the clock pulses to the clock input of the 3rd flip-flop. The 3rd flip-flop toggles state with every fourth negative clock transition at d and h on the time line.

The wave form and the truth table show that the synchronous counter progresses upward in a natural binary sequence from 000 to 111. The total count from 000 to 111 is 8 and hence this counter can also be called MOD-8 counter, in count up mode.



CHECK YOUR PROGRESS

Q14. State True or False:

a) Counters are non sequential digital circuits.

b) Asynchronous counters are faster in operation than synchronous counters.

c) The natural progression of a counter is called MODE.

d) Counters can be used to build digital clock.

2.7 REGISTER

A number of flip-flop connected to store binary number is called a Register. The number to be stored is entered or shifted into the register and also taken out or shifted out as per necessity. Hence, registers are also known as shift register.

Registers are used to store data temporarily. Registers can be used to perform some important arithmetic operations like complementation, multiplication, division etc. It can be connected to form counters, to convert serial data to parallel and parallel to serial data.

Types of registers: According to shifting of binary number different types of registers are:

- Serial In—Serial Out(SISO)
- Serial In –Parallel Out (SIPO)
- Parallel In –Serial Out(PISO)
- Parallel In –Parallel Out(PIPO)

2.7.1 Serial In – Serial Out Register (SISO)

Figure 2.24 shows a typical 4 bit SISO register using flip-flops. Here the content of the register is named as QRST. Let us consider that all flip-flops are initially reset. Hence at the beginning QRST= 0000. Let us consider a binary number 1011 which we want to store in the SISO register.

At time A: A 1 is applied at the D input at the first flip-flop. At the negative edge of the CLK pulse, this 1 is shifted into Q. The O of Q is shifted into R, O of R is shifted into S and O of S is shifted into T. The output of flip-flop just after time A is QRST=1000.

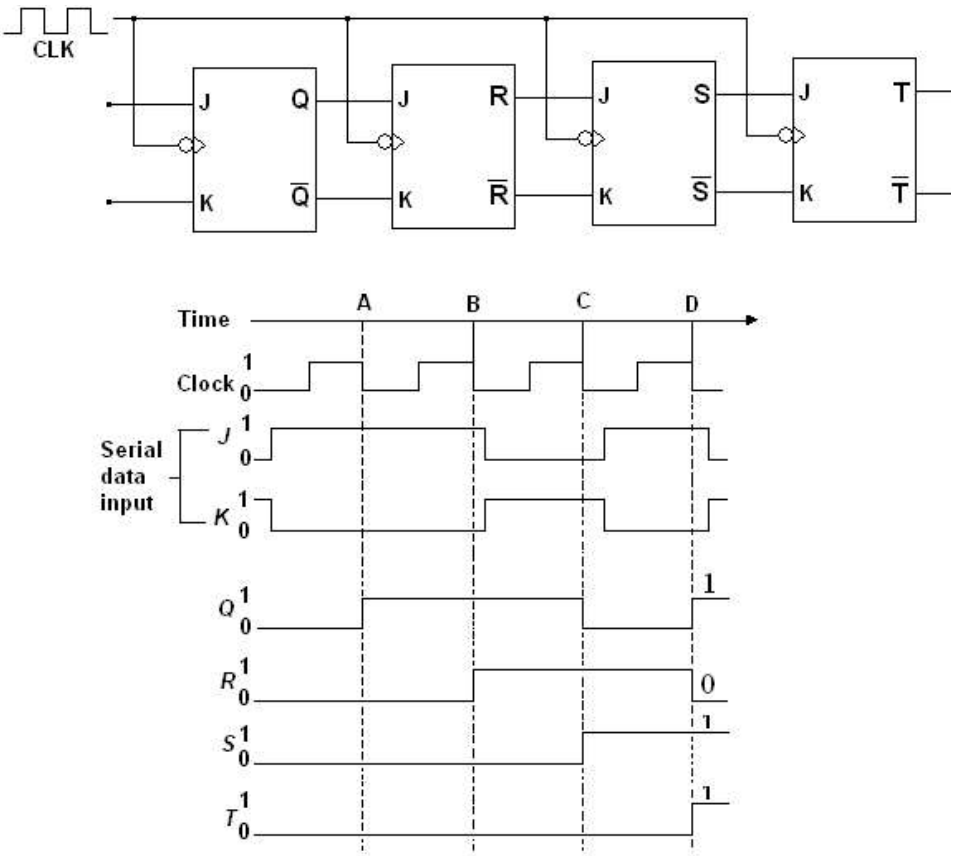


Figure 2.24: 4 Bit Serial In – Serial Out Shift Register

At time B: Another 1 is applied in the data input of the first flip-flop. So at the negative CLK edge, this 1 is shifted to Q. The 1 of Q is shifted in R.,O of R shifted in S, O of S is shifted into T. so, at the end of time B the output of all the flip-flops is QRST=1100.

At time C: A O(zero) is applied in the D input of the 1st first flip-flop. At the negative CLK edge, this O shifts to Q. The 1 of Q shifts into R.,1 of R shifts into S, 1 of R shifts into S, O of S shifts into T. Hence, the output becomes QRST=0110.

At time D: 1 is applied to D input of the first flip-flop. So this 1 shifts into Q at the negative transmission of CLK. The previous O of Q shifts into R, the 1 of R shifts into S, the 1 of S shifts into T. Hence at the end of time D, the registers contains QRST=1011.

In the above steps, using 4 CLK pulses, we have shifted a 4 bit binary number 1011 in the register in a serial fashion.

To take out this binary number serially, we need another 4 CLK pulses and 4 O inputs into D pin of the first flip-flop. The binary number leave the register serially through the T pin of the last flip-flop.

2.7.2 Serial In—Parallel Out (SIPO)

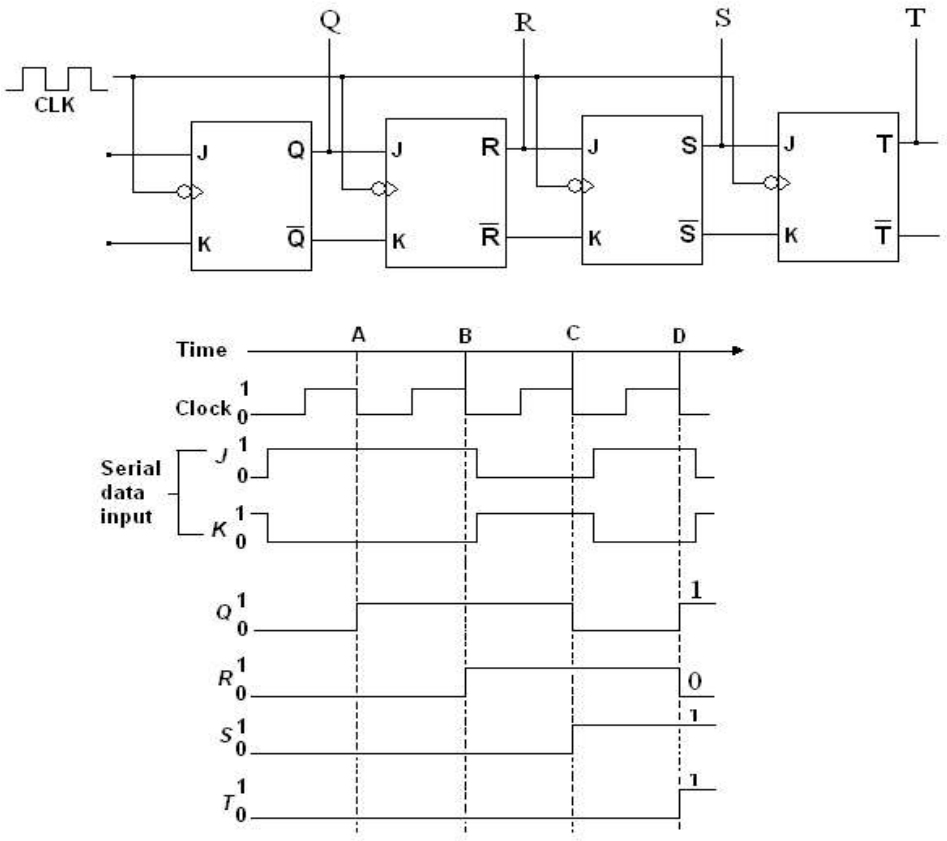


Figure 2.25: 4 Bit Serial In – Parallel Out Shift Register

In this type of shifts register, data are entered serially into the register and once data entry is completed it can be taken out parallelly. To take the data parallelly, it is simply required to have the output of each flip-flop to an output pin. All other constructional features are same as Serial In—Serial Out (SISO) register.

The shifting of data into SIPO is same as SISO registers. In the SIPO of Figure 2.25, a binary number, say, 1011 would be shifted just like the manner as described in the previous section. It would take 4 CLK pulses to complete the shifting. As soon as shifting is completed, the stored binary number becomes available in the output pins QRST. SIPO register is useful to convert serial data into parallel data.

2.7.3 Parallel In-Serial Out Register (PISO)

PISO register takes data parallel i.e. data are stored at a time and shifts data serially. Commercially available TTL IC for PISO is 54/74166. To understand the functional block diagram of 54/74166 we should first understand the following-

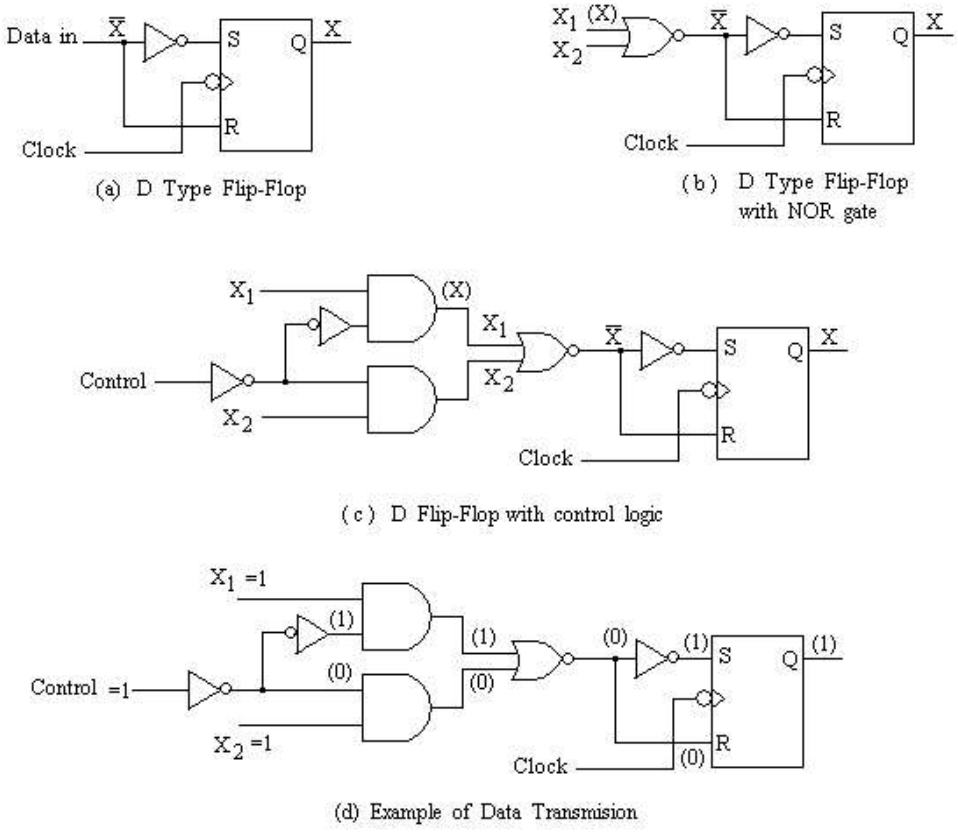
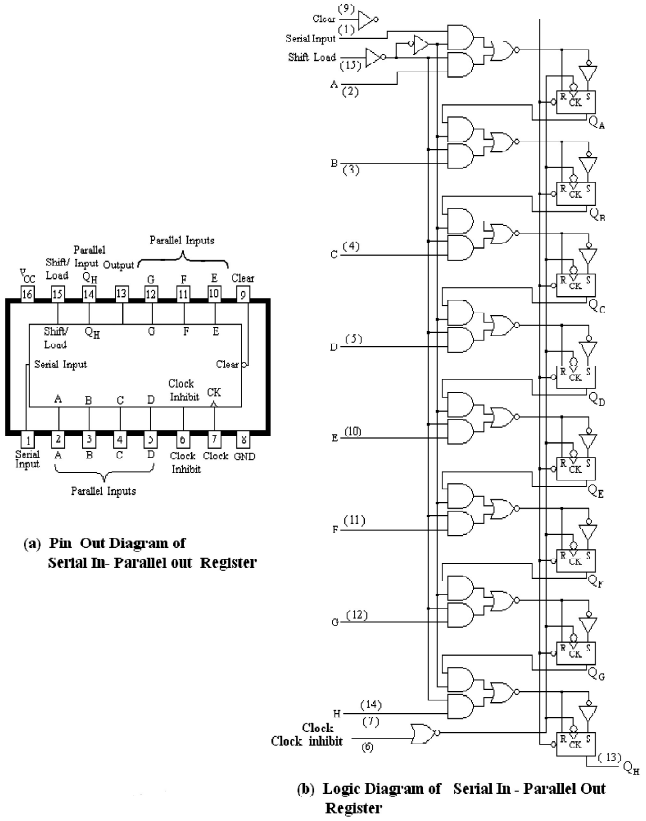


Figure 2.26: Building Block of Parallel In-Serial Out Register

Figure 2.26(a) is a clocked RS flip-flop, which is converted to D flip-flop by a NOT gate. The output of the flip-flop is 1 if Data IN ($\overline{X}$) is 0. Next add a NOR gate as in Figure 2.26(b). Here, if X_2 is at ground level, X_1 will be inverted by the NOR gate. As for example, if =1, then output of the NOR gate will be =0, thereby a 1 will be applied as S input into the flip-flop. This NOR gate allows entering data from two sources, either from or . To shift into the flip-flop, is kept at ground level and to shift into the flip-flop, is kept at ground level. Here, ground level connection implies a 0(zero) input.

Now in Figure 2.26(c) two AND gates and two NOT gates are added. These will allow the selection of data or data . If the control line is high, the upper AND gate is enabled and the lower AND gate is disabled. Thus, the data will enter at the upper leg of the NOR gate and at the same time the lower leg of the NOR gate is kept at ground. Opposite to this, if the control is low, the upper AND gate is disabled and the lower AND gate is enabled. So will appear at the lower leg of the NOR gate and during this time the upper leg of the NOR leg gate is kept at ground level.



(a) Pin Out Diagram of Serial In- Parallel out Register

(b) Logic Diagram of Serial In - Parallel Out Register

Figure 2.27: Circuit of 54/74166 (a) Pin Out Diagram of Serial In-Parallel Out Register (b) Logic Diagram of Serial In-Parallel Out Register

If we study the figure 2.27 of PISO we see that circuit of figure 2.26 (a) is repeated 8 times to form the 54/74/66 shift register. These 8 circuits are connected in such a style that it allows two operations: (1) The parallel data entry and (2) shifting of data serially through the flip-flop from Q_A toward Q_B

If figure 2.27 the X_2 input of figure 2.26(c) is taken out from each flip-flop to form 8 inputs named as ABCDEFGH to enter 8 bit data parallely to the register. The control is named here as SHIFT/LOAD which is kept

low to load 8 bit data into the flip-flops with a single clock pulse parallely. If the SHIFT/LOAD is kept high it will enable the upper AND gate for each flip-flop. If any input is given to this upper AND gate then a clock pulse will shift a data bit from one flip-flop to the next flip-flop. That means data will be shifted serially.

2.7.4 Parallel In –Parallel Out Register (PIPO)

The register of Figure 2.27 can be converted to PIPO register simply by adding an output line from each flip-flop.

The 54/74198 is an 8 bits such PIPO and 54/7459A is a 4 bit PIPO register. Here the basic circuit is same as Figure 2.26 ©. The parallel data outputs are simply taken out from the Q sides of each flip-flop. In Figure 2.28 the internal structure of 54/7459A is shown.

When the MODE CONTROL line is high, the data bits ABCD will be loaded into the register parallely at the negative clock pulse. At the same time the output is available $Q_A Q_B Q_C Q_D$. When the MODE CONTROL is low, then the left AND gate of the NOR gate is enabled. Under this situation, data can be entered to the register serially through SERIAL INPUT. In each negative transition, a data bit shifted serially from Q_A to Q_B, from Q_B to Q_C and so on. This operation is called right-shift operation.

With a little modification of the connection, the same circuit can be used for shift-left operation. To operate in shift-left mode, the input data is to be entered through D input pin. It is also necessary to connect Q_D to C, Q_C to B, Q_B to A as shown in Figure 2.28. MODE CONTROL line is high to enter data through the D input pin and each stored data bits of flip-flops will be shifted to left flip-flop on each negative clock transition. This is serial data and left shift operation.

To clock inputs—clock1 and clock2 is used here to perform shift right and shift left operation..

Hence 54/7495A can be used as Parallel In –Parallel Out shift register as well as shift right and shift left register.

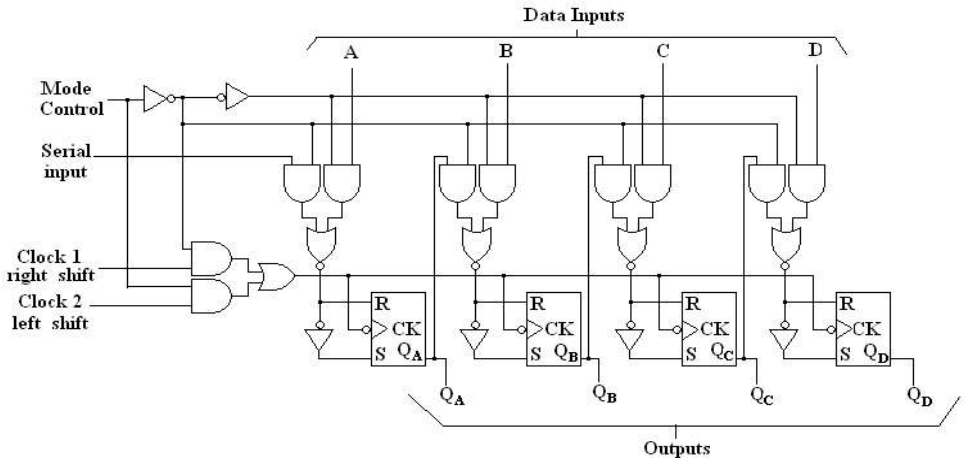


Figure 2.28: Parallel In- Parallel Out Shift Register



CHECK YOUR PROGRESS

Q15. Shift registers are_____.

a) basically a synchronous circuit

b) an asynchronous circuit

c) permanent memory

d) none of these

Q16. In SIPO _____.

a) data enters parallely and leaves serially

b) data enters serially and leaves serially

c) data enters serially and leaves parallely

d) data enters parallely and leaves parallely



2.8 LET US SUM UP

- Digital circuits are of two categories - combinational and sequential
- A combinational circuit is some combinations of logic gates as per specific relationship between inputs and outputs.

- Adder and subtractor circuits can perform binary addition and subtraction.
- A multiplexer is a combinational circuit which selects one of many inputs.
- Demultiplexer is opposite to a multiplexer.
- An encoder generates a binary code for 2^n input variables.
- A decoder decodes an information received from n input lines and transmits the decoded information to maximum outputs.
- A sequential circuit's output depends on past output and present inputs.
- A flip-flop is basically a single cell of memory which can store either 1 or 0.
- Sequential circuits use flip-flop as their building block.
- There are many types of flip-flop viz RS, D, JK, MS flip-flop.
- A counter is a sequential circuit that can count square waves give as clock input. There are two types of counters- asynchronous and synchronous counter.
- Shift registers are also sequential circuit which are used to store binary bits. They are of four different types - Serial In- Serial Out, Serial In- Parallel Out, Parallel In- Parallel Out and Parallel In-Serial Out register.



2.9 FURTHER READINGS

1) Mano, M. M. (2006). Computer systems architecture.
2) Mano, M. (1979). Digital logic. Computer Design. Englewood Cliffs Prentice-Hall.
3) Talukdar, P.H. Digital Techniques. N. L. Publications.
4) Lee, S. C. (1976). Digital circuits and logic design.
5) Leach, D. P., & Malvino, A. P. (1994). Digital Principles and Applications. Glencoe/McGraw-Hill.

2.10 ANSWER TO CHECK YOUR PROGRESS

Ans to Q No 1: (b)

Ans to Q No 2: (c)

Ans to Q No 3: (a)

Ans to Q No 4: (b)

Ans to Q No 5: (c)

Ans to Q No 6: (d)

Ans to Q No 7: (b)

Ans to Q No 8: (a)

Ans to Q No 9: (b)

Ans to Q No 10: (a)

Ans to Q No 11: (c)

Ans to Q No 12: (b)

Ans to Q No 13: (c)

Ans to Q No 14: (a) False (b) False (c) True (d) True

Ans to Q No 15: (a)

Ans to Q No 16: (c)

2.11 MODEL QUESTIONS

Q1. Distinguish between combinational circuit and sequential circuit.

Q2. With truth table and logic diagram explain the working of a full-adder circuit.

Q3. With truth table and logic diagram explain the working of a full-subtractor circuit.

Q4. What is a multiplexer? With diagram explain the working of a 8-to-1 multiplexer.

Q5. Explain the principle of an encoder. Draw a decimal-to-BCD encoder.

Q6. What are the differences between asynchronous and synchronous counter? Draw a MODE-8 counter and explain its working principle.

Q7. Draw logic diagram with output wave form of a 4-bit Serial In-Parallel Out shift register for an input of 1101. Explain its operation.

Q8. Why is square wave clock pulse converted to a narrow spike to be used for flip-flops? Draw a RC differentiator circuit to convert a square wave into a narrow spike.

Q9. What is racing? To get rid of racing what techniques are used?

Q10. What is a magnitude comparator? Draw a block diagram and the function table of the magnitude comparator SN 7485.