

Q.2 a. What is a Digital System? Explain the advantages and limitations of Digital Techniques over Analog Techniques.

Answer:

Digital System: A digital system is a combination of devices designed to manipulate logical information or physical quantities that are represented in digital form i.e., the quantities can take on only discrete values.

Advantages of Digital Techniques over Analog Techniques:

- (i) **Digital systems are generally easier to design:** The circuits used in digital systems are switching circuits, where exact values of voltage or current are not important, only the range (HIGH or LOW) in which they fall.
- (ii) **Information storage is easy:** This is accomplished by special devices and circuits that can latch onto digital information and hold it for as long as necessary and mass storage techniques that can store billions of bits of information in a relatively small physical space. Analog storage capabilities are extremely limited.
- (iii) **Accuracy and precision are easier to maintain throughout the system:** Once a signal is digitized, the information it contains does not deteriorate as it is processed. In analog systems, the voltage and current signals tend to be distorted by the effects of temperature, humidity and component tolerance variations in the circuits that process the signal.
- (iv) **Operation can be programmed:** It is fairly easy to design digital systems whose operation is controlled by a set of stored instructions called a program. Analog systems can also be programmed, but the variety and the complexity of the available operations are severely limited.
- (v) **Digital circuits are less affected by noise:** Spurious fluctuations in voltage (noise) are not as critical in digital systems because the exact value of a voltage is not important, as long as the noise is not large enough to prevent us from distinguishing a HIGH from a LOW.
- (vi) **More digital circuitry can be fabricated on IC chips:** More amount of digital circuitry are fabricated on IC chip, whereas analog circuitry has complexity with the use of devices that cannot be economically integrated (high value capacitors, precision resistors, inductors, transformers) have prevented analog systems from achieving the same high degree of integration.

Limitations of Digital Techniques over Analog Techniques:

- (i) The real world is analog.
- (ii) Processing digitized signals takes time.

b. Convert the decimal number 82.67 to its equivalent binary number.

Answer:

Conversion of Decimal number 82.67 to its Binary Equivalent

Considering the integer part 82 and finding its binary equivalent

2	82	
2	41	Remainder ----- 0 (LSB)
2	20	Remainder ----- 1
2	10	Remainder ----- 0
2	5	Remainder -----0
2	2	Remainder ----- 1
2	1	Remainder ----- 0
	0	Remainder ----- 1 (MSB)

The Binary equivalent is $(1010010)_2$

Now taking the fractional part i.e., 0.67

Fraction	Fraction X 2	Remainder New Fraction	Integer
0.67	1.34	0.34	1
0.34	0.68	0.68	0
0.68	1.36	0.36	1
0.36	0.72	0.72	0
0.72	1.44	0.44	1
0.44	0.88	0.88	0
0.88	1.76	0.76	1
0.76	1.52	0.52	1

It is seen that, it is not possible to get a zero as remainder even after 8 stages. The process continued further on an approximation can be made and the process is terminated here.

The binary equivalent is 0.10101011

Therefore, the binary equivalent of decimal number **82.67** is **$(1010010.10101011)_2$**

- Q.3 a. Simplify the logic expression $F = \overline{ABC} + \overline{AB}C + \overline{A}B\overline{C} + A\overline{B}C + A\overline{B}C$ using boolean algebraic theorems.**

Answer:

Simplification of the logic expression $F = \overline{ABC} + \overline{AB}C + \overline{A}B\overline{C} + A\overline{BC} + A\overline{B}C$

$$F = \overline{ABC} + \overline{AB}C + \overline{A}B\overline{C} + A\overline{BC} + A\overline{B}C$$

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

$$(\because \overline{ABC} = \overline{A+B+C} \text{ and } \overline{AB} = \overline{A+B} \text{ by using Demorgan's Laws})$$

$$= \overline{A+B+C} + \overline{A}C + \overline{B}C + \overline{AB}C + A\overline{B} + A\overline{C} + A\overline{B}C$$

$$= \overline{A} + \overline{A}C + \overline{B} + \overline{B}C + \overline{C} + A\overline{C} + \overline{AB}C + A\overline{B} + A\overline{B}C$$

$$= \overline{A}(1+C) + \overline{B}(1+C) + \overline{C}(1+A) + \overline{AB}C + A\overline{B} + A\overline{B}C$$

$$= \overline{A} + \overline{B} + \overline{C} + \overline{AB}C + A\overline{B} + A\overline{B}C \{ \because (1+C) = 1 \text{ and } (1+A) = 1 \}$$

$$= (\overline{A} + A\overline{B}) + \overline{B}(1+AC) + \overline{C}(1+A\overline{B})$$

$$= (\overline{A+B}) + \overline{B} + \overline{C} \{ \because (\overline{A} + A\overline{B}) = (\overline{A+B}); (1+AC) = 1 \text{ and } (1+A\overline{B}) = 1 \}$$

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

b. Draw the logic diagram for 4-bit Even Parity Generator and explain its operation.

Answer:

4-bit Even Parity Generator:

Fig. (1) show the logic diagram of 4-bit Even parity Generator, where a group of four bits as the data to be transmitted. The set of data to be transmitted is applied to the parity-generator circuit, which produces the even-parity bit, P at its output. The transmitted from the binary number from 0000 to 1111 and its parity bit is shown in the truth table of Table.1. The transmitted data is the summation of transmitted data and parity bit is also shown in Table.1. This parity bit

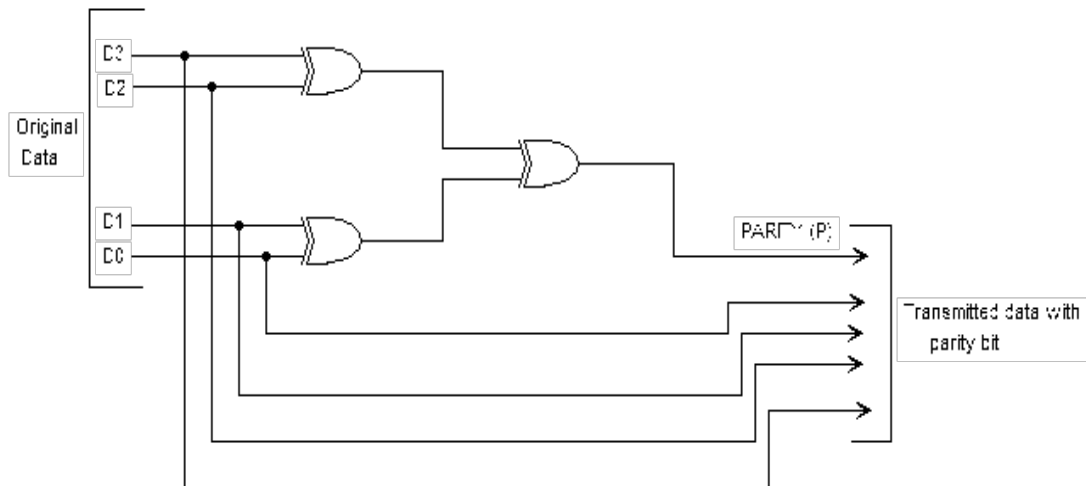


Fig.1 EVEN PARITY GENERATOR

is transmitted to the receiver along with the original data bits, making a total of five bits.

S.No.	Transmitted Data	Parity Bit (P)	Transmitted Data with Parity Bit
1	0000	0	00000
2	0001	1	00001
3	0010	1	00101
4	0011	0	00110
5	0100	1	01001
6	0101	0	01010
7	0110	0	01100
8	0111	1	01111
9	1000	1	10001
10	1001	0	10010
11	1010	0	10100
12	1011	1	10111
13	1100	0	11000
14	1101	1	11011
15	1110	1	11101
16	1111	0	11110

Table.1 Truth Table for Even Parity Generator

- c. Minimize the logic function $F(A, B, C, D) = \sum m(1,3,5,8,9,11,15) + d(2,13)$ using K-maps.

Answer:

Minimization of the logic function $F(A, B, C, D) = \sum m(1,3,5,8,9,11,15) + d(2,13)$

- (i) Karnaugh Map for the logic function is given in table 2

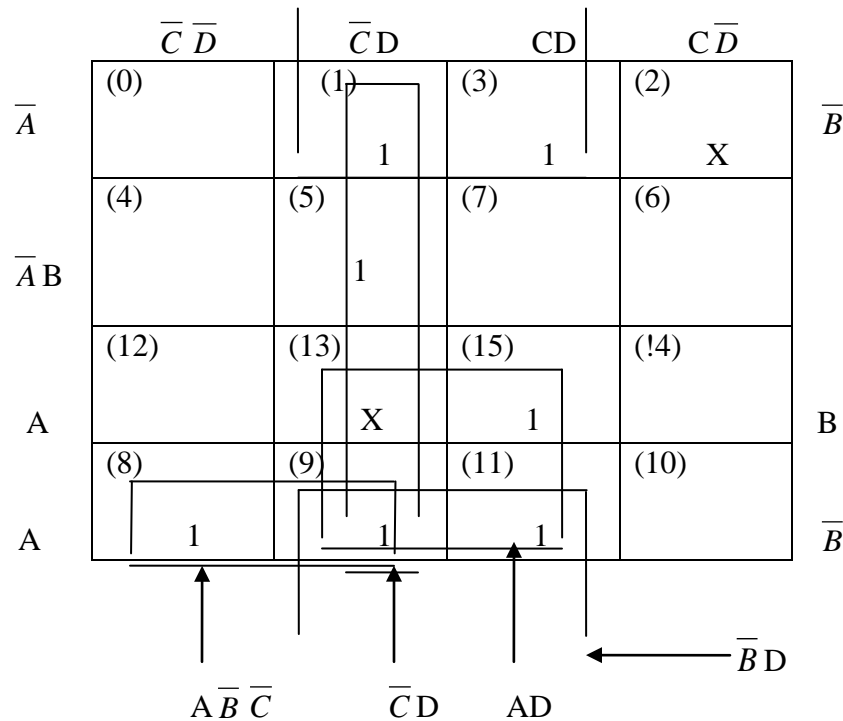


Table 2

The minimized logic expression in SOP form is $F = A \bar{B} \bar{C} + \bar{C} D + \bar{B} D + AD$

The minimized logic expression in POS form is $F = (A + \bar{B} + \bar{C}) (\bar{C} + D) (\bar{B} + D) (A + D)$

Q.4 a. What is a Flip-Flop? Draw the logic diagram for Master Slave Flip-Flop and explain its function with the help of truth table.

Answer:

Flip-Flop: A flip-flop is a basic memory element used to store one bit of information. Both Flip-flops and latches are bistable logic circuits and can reside in any of the two stable states due to a feedback arrangement. The main difference between them is in the method used for changing the state.

Logic Diagram of Master-Slave J-K Flip-Flop using NAND Gates: Fig.2 shows the logic diagram of Master-Slave J-K Flip-Flop using NAND gates.

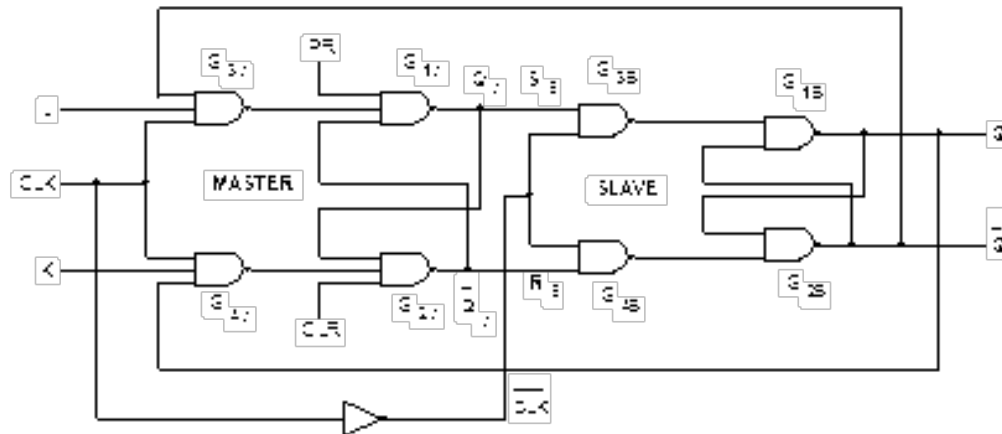


Fig.2 Logic Diagram of Master-Slave J-K FLIP-FLOP

The Race-around Condition: The difficulty of both inputs 1 ($S = R = 1$) being not allowed in an S - R Flip-Flop is eliminated in a J - K Flip-Flop by using the feedback connection from outputs to the inputs of the gates. In R - S Flip-Flop, the inputs do not change during the clock pulse ($CK = 1$), which is not true in J - K Flip-Flop because of the feedback connections. Consider that the inputs are $J = K = 1$ and $Q = 0$ and a pulse as shown in Fig.3 is applied at the clock input. After a time interval Δt equal to the propagation delay through two NAND gates in series, the output will change to $Q = 1$.

Now we have $J = K = 1$ and $Q = 1$ and after another time interval of Δt the output will change back to $Q = 0$. Hence, for the duration t_p of the clock pulse, the output will oscillate back and forth between 0 and 1. At the end of the clock pulse, the value of Q is uncertain. This situation is referred to as the race-around condition. The race-around condition can be avoided if $t_p < \Delta t < T$. However, it may be difficult to satisfy this inequality because of very small propagation delays in ICs. A more practical method of overcoming this difficulty is the use of the master-slave (M - S) configuration.

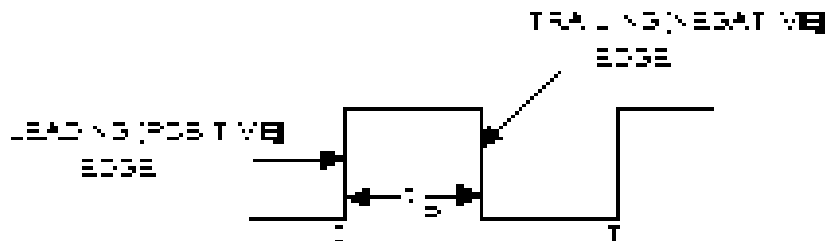


Fig.3 a Clock Pulse

A master-slave J - K Flip-Flop is a cascade of two S - R Flip-Flops with feedback from the outputs of the second to the inputs to the first as illustrated in Fig.2. Positive clock pulses are applied to the first Flip-Flop and the clock pulses are inverted before these are applied to the second Flip-Flop. When $CK=1$, the first Flip-Flop is enabled and the outputs Q_M and $\overline{Q_M}$ respond to their inputs J and K according to the Table 7.1. At this time, the second Flip-Flop is inhibited because its clock is LOW ($\overline{CK} = 0$). When CK goes LOW ($\overline{CK} = 1$), the first Flip-Flop is inhibited and the second Flip-Flop is enabled, because now its

clock is HIGH ($\overline{CK} = 1$). Therefore, the outputs Q and \overline{Q} Follow the outputs Q_M and $\overline{Q_M}$ respective (second and third rows of Table 3). Since the second Flip-Flop simply follows the first one, it is referred to as the Slave and the first one as the Master. Hence, this configuration is referred to as Master-Slave Flip-Flop. In this circuit, the inputs to the gates G_{3M} and G_{4M} do not change during the clock pulse, therefore the Race-around condition does not exist. The state of the Master-Slave Flip-Flop changes at the negative transition (trailing end).

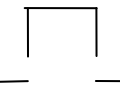
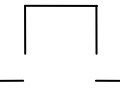
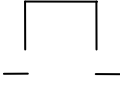
Inputs					Output
PR	CLR	CLK	J	K	Q
0	0	X	X	X	Race Condition
0	1	X	X	X	1
1	0	X	X	X	0
1	1	X	0	0	No change
1	1		0	1	0
1	1		1	0	1
1	1		1	1	Toggle

Table 3 Truth Table of JK Master-Slave Flip-Flop

- b. Explain the application of Flip-Flop as a Shift Register using D Flip-Flops.**

Answer:

Application of Flip-Flop as a Shift Register (SERIAL IN - SERIAL OUT SHIFT REGISTER) using D Flip-Flops;

4 shows a 4 bit serial in - serial out shift register consisting of four D flip flops FF_0 , FF_1 , FF_2 and FF_3 . As shown it is a positive edge triggered device. The working of this register for the data 1010 is given in the following steps.

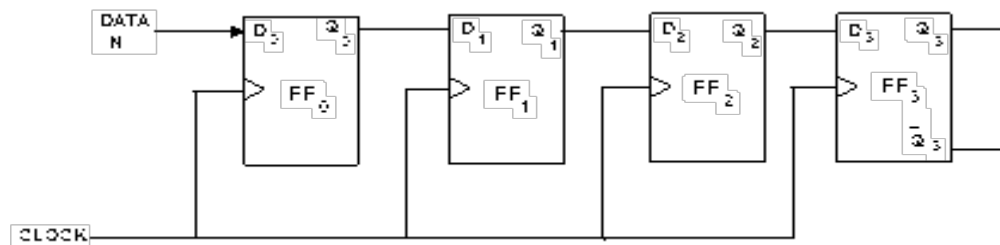


Fig.4 Logic Diagram of 4-bit Serial In – Serial Out Shift Register

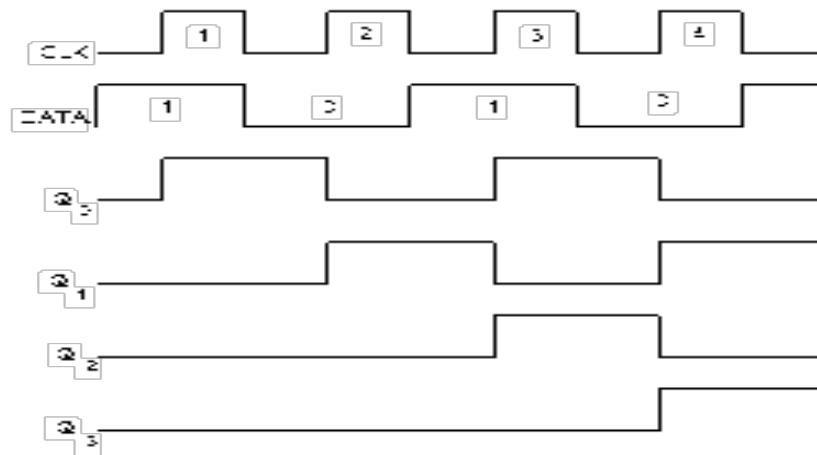


Fig.5 Output Waveforms of 4-bit Serial-in Serial-out Register

1. Bit 0 is entered into data input line. $D_0 = 0$, first clock pulse is applied, FF_0 is reset and stores 0.
2. Next bit 1 is entered. $Q_0 = 0$, since Q_0 is connected to D_1 , D_1 becomes 0.
3. Second clock pulse is applied, the 1 on the input line is shifted into FF_0 because FF_0 sets. The 0 which was stored in FF_0 is shifted into FF_1 .
4. Next bit 0 is entered and third clock pulse applied. 0 is entered into FF_0 , 1 stored in FF_0 is shifted to FF_1 and 0 stored in FF_1 is shifted to FF_2 .
5. Last bit 1 is entered and 4th clock pulse applied. 1 is entered into FF_0 , 0 stored in FF_0 is shifted to FF_1 , 1 stored in FF_1 is shifted to FF_2 and 0 stored in FF_2 is shifted to FF_3 . This completes the serial entry of 4 bit data into the register. Now the LSB 0 is on the output Q_3 .
6. Clock pulse 5 is applied. LSB 0 is shifted out. The next bit 1 appears on Q_3 output.
7. Clock pulse 6 is applied. The 1 on Q_3 is shifted out and 0 appears on Q_3 output.
8. Clock pulse 7 is applied. 0 on Q_3 is shifted out. Now 1 appears on Q_3 output.
9. Clock pulse 8 is applied. 1 on Q_3 is shifted out.
10. When the bits are being shifted out (on CLK pulse 5 to 8) more data bits can be entered in.

Fig.5 shows the output waveforms of 4-bit Serial-in Serial-out Register.

- Q.5 a. (i) Perform the Addition of -20 to +26 using 2's complement System.
 (ii) Perform the Subtraction of 0011.1001- 0001.1110 using 2's Complement System.

Answer:

(i) Addition of -20 to +26 using 2's complement:

First convert the two numbers 20 and 26 into its 8-bit binary equivalent and find out the 2's complement of 20, then add -20 to +26.

$$\begin{array}{r}
 20 = 00010100 \text{ (8-bit binary equivalent of 20)} \\
 \overline{20} = 11101011 \text{ (1's complement)} \\
 \quad \quad \quad +1 \\
 \hline
 \overline{20} = -20 = 11101100 \text{ (2's complement of 20)} \\
 +26 = 00011010 \text{ (8-bit binary equivalent of 26)} \\
 \hline
 \text{Addition of -20 to +26} = +6 = 00000110 \\
 \hline
 \end{array}$$

Hence -20 to +26 = (6)₁₀ = (0110)₂

(ii) Subtraction of 0011.1001 – 0001.1110:

1's complement of 0001.1110 is 1110.0001 and
 its 2's complement is 1110.0010.

$$\begin{array}{r}
 0011.1001 = 0011.1001 \\
 - 0001.1110 = + 1110.1011 \text{ (2's complement)} \\
 \hline
 \end{array}$$

$$\begin{array}{r}
 1 \ 0001.1011 = + 1.68625 \\
 \uparrow \\
 \text{Ignore}
 \end{array}$$

If a final carry is generated discard the carry and the answer is given by the remaining bits which is positive i.e., (0001.1011)₂ = (+ 1.68625)₁₀

b. What is the need of Parallel Binary Adder? Draw the block diagram of four-bit Parallel Adder using Full Adders and explain its operation.

Answer:

Binary Adder: The full adder forms the sum of two bits and a previous carry. Two binary numbers of n bits each can be added by means of Binary Adder. If A = 1011 and B = 0011, whose sum is S = 1110. When pair of bits is added through a full-adder, the circuit produces a carry to be used with the pair of bits one significant position higher. This is shown in Table 4

The bits are added with full-adders, starting from the Least Significant Position (subscript 1), to form the sum bit and carry bit. The input carry C_1 in the Least Significant position must be 0. The value of C_{i+1} in a given significant position is the output carry of the full-adder. This value is transferred into the input carry of the full-adder that adds the bits one higher significant position to the left. The sum bits are thus generated starting from the rightmost position and are available as soon as the corresponding previous carry bit is generated.

Subscript i	4	3	2	1		Full-Adder
Input Carry	0	1	1	0	C_i	Z
Augend	1	0	1	1	A_i	X
Addend	0	0	1	1	B_i	Y
Sum	1	1	1	0	S_i	S
Output Carry	0	0	1	1	C_{i+1}	C

Table 4 Truth Table for Binary Adder

A Binary Parallel Adder is a digital function that produces the arithmetic sum of two binary numbers in parallel. It consists of full-adders connected in cascade, with the output carry from one full-adder connected to the input carry of the next full-adder. Fig.6 shows a 4-bit Binary Parallel Adder. The augends bits of A and the addend bits of B are designated by subscript numbers from right to left, with subscript 1 denoting the low-order bit. The carries are connected in a chain through the full-adders. The input carry to the adder is C_1 and the output carry is C_5 . The S outputs generate the required sum bits.

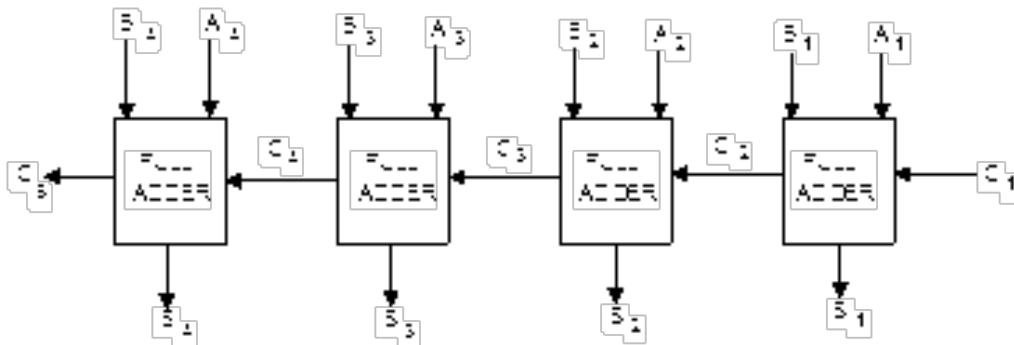


Fig.6 4-bit Binary Parallel Adder using Full-Adders

Q.6 a. What is a Ripple Counter? Draw the logic diagram of 3-bit Ripple Counter and explain its working with the help of timing diagram.

Answer:

Ripple Counter: In Ripple Counters, all the Flip-Flops are not clocked simultaneously and the flip-flops do not change state exactly at the same time.

3-bit Ripple Counter: A 3-bit Binary Counter has maximum of 2^3 states i.e., 8 states, which requires 3 Flip-Flops. The word Binary Counter means a counter which counts and produces binary outputs 000,001,010-----111. It goes through a binary sequence of 8 different states (i.e., from 0 to 7).

Fig.7 shows the logic circuit of a 3-bit Binary Ripple Counter consisting of 3 Edge Triggered JK flip-flops. As indicated by small circles at the CLK input of flip-flops, the triggering occurs when CLK input gets a negative edge. Q_0 is the Least Significant Bit (LSB) and Q_2 is the Most Significant Bit (MSB). The flip-flops are connected in series. The Q_0 output is connected to CLK terminal of second flip-flop. The Q_1 output is connected to CLK terminal of third flip-flop. It is known as a Ripple Counter because the carry moves through the flip-flops like a ripple on water.

Working: Initially, CLR is made Low and all flip-flops Reset giving an output $Q = 000$. When CLR becomes High, the counter is ready to start. As LSB receives its clock pulse, its output changes from 0 to 1 and the total output $Q = 001$. When second clock pulse arrives, Q_0 resets and carries (i.e., Q_0 goes from 1 to 0 and, second flip flop will receive CLK input). Now the output is $Q = 010$. The third CLK pulse changes Q_0 to 1 giving a total output $Q = 011$. The fourth CLK pulse causes Q_0 to reset and carry and Q_1 also resets and carries giving a total output $Q = 100$ and the process goes on. The action is shown in Table 8.1. The number of output states of a counter are known as Modulus (or Mod). A Ripple Counter with 3 flip-flops can count from 0 to 7 and is therefore, known as Mod-8 counter.

Counter State	Q_2	Q_1	Q_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Table.5 Counting Sequence of a 3-bit Binary Ripple Counter

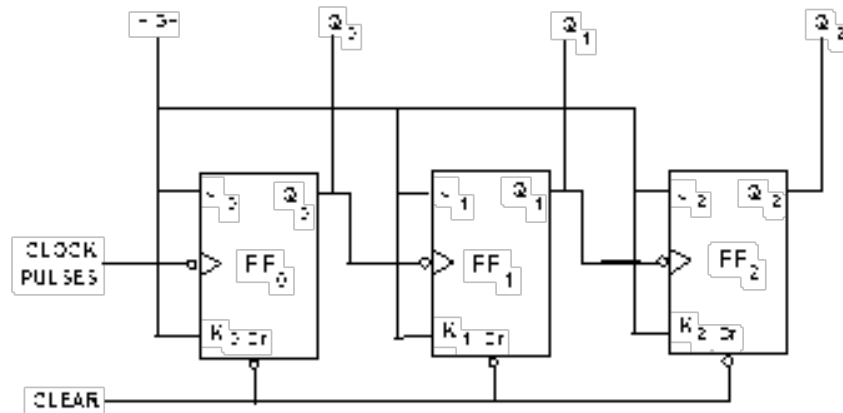


Fig.7 Logic Diagram of 3-Bit Binary Ripple Counter

Ripple counters are simple to fabricate but have the problem that the carry has to propagate through a number of flip flops. The delay times of all the flip flops are added. Therefore, they are very slow for some applications. Another problem is that unwanted pulses occur at the output of gates.

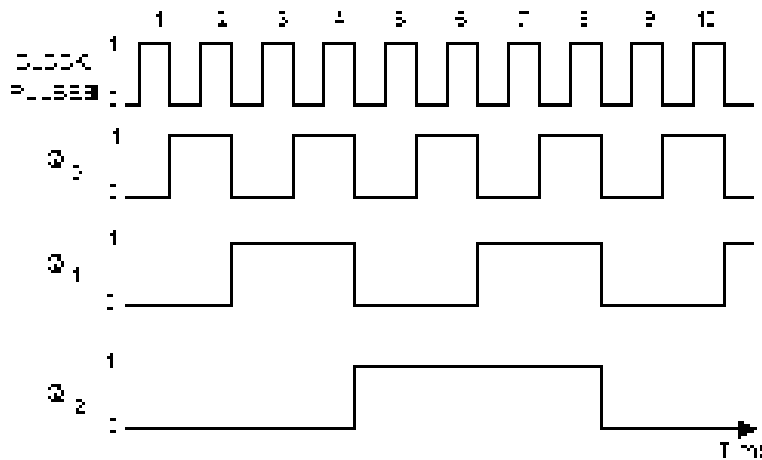


Fig.8 Timing Diagram of 3-bit Binary Ripple Counter

The timing diagram is shown in *Fig.8*. FF_0 is LSB flip flop and FF_2 is the MSB flip flop. Since FF_0 receives each clock pulse, Q_0 toggles once per negative clock edge as shown in *Fig. 8(b)*. The remaining flip flops toggle less often because they receive negative clock edge from preceding flip flops. When Q_0 goes from 1 to 0, FF_1 receives a negative edge and toggles. Similarly, when Q_1 changes from 1 to 0, FF_2 receives a negative edge and toggles. Finally when Q_2 changes from 1 to 0, FF_3 receives a negative edge and toggles. Thus whenever a flip flop resets to 0, the next higher flip flop toggles.

This counter is known as ripple counter because the 8th clock pulse is applied, the trailing edge of 8th pulse causes a transition in each flip flop. Q_0 goes from High to Low, this

causes Q_1 go Low. Thus the effect ripples through the counter. It is the delay caused by this ripple which results in a limitation on the maximum frequency of the input signal.

- b. **What is Synchronous Counter? Draw the logic circuit of Mod-8 Synchronous Counter and explain its working with timing waveform.**

Answer:

(b) Synchronous Counters: The term synchronous means that all flip-flops are clocked simultaneously. The clock pulses drive the clock input of all the flip-flops together so that there is no propagation delay.

MOD 8 SYNCHRONOUS COUNTER: In a synchronous counter, where all flip-flops are clocked simultaneously and the clock pulses drive the clock input of all the flip-flops together so that there is no propagation delay. Fig. 9 shows the circuit of Mod 8 Synchronous Counter. In this case the D input of FF_0 is driven by $\overline{Q_3}$ output of FF_3 , i.e., the complement of the output of the last flip flop is fed to the D of FF_0 . This feedback arrangement produces the sequence of states shown in Table 6. The 4 bit sequence has a total of 8 states (a n bit sequence will have 2^n states). Thus an n bit synchronous counter will have a modulus of 2^n .

The Q output of each stage feeds the D input of next stage. But the \overline{Q} output of the last stage feeds the D input of first stage. The counter fills up with 1s from left to right and then fills up 0s again as shown in Table 6.

Fig. 10 shows the wave shapes/timing diagram of Mod 8 synchronous counter.

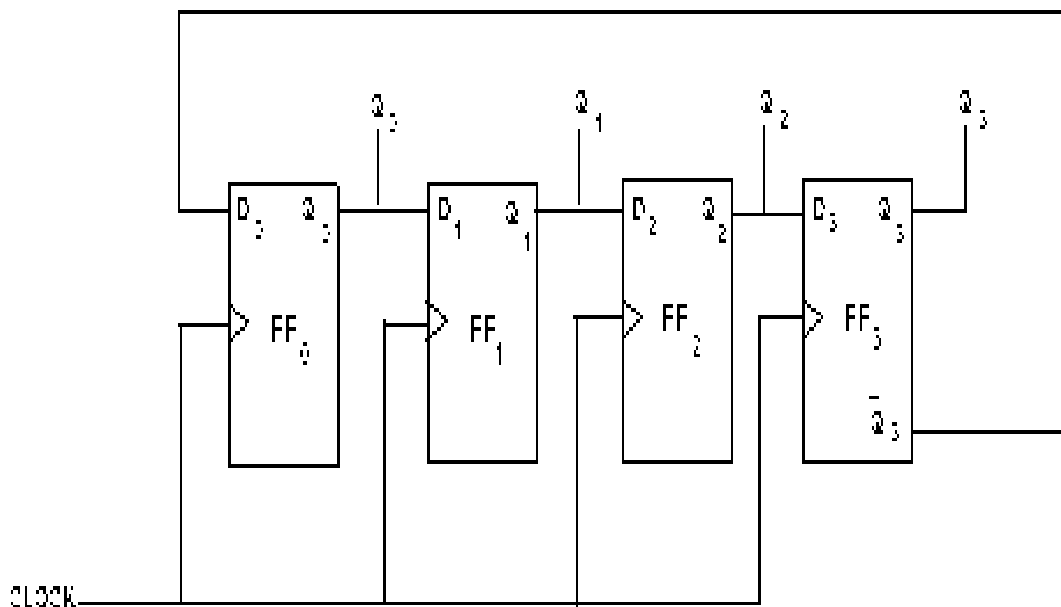


Fig.9 Logic Diagram of MOD 8 Synchronous Counter

Clock Pulse	Q ₀	Q ₁	Q ₂	Q ₃
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1

Table 6 Sequence of states of Mod 8 Synchronous Counter

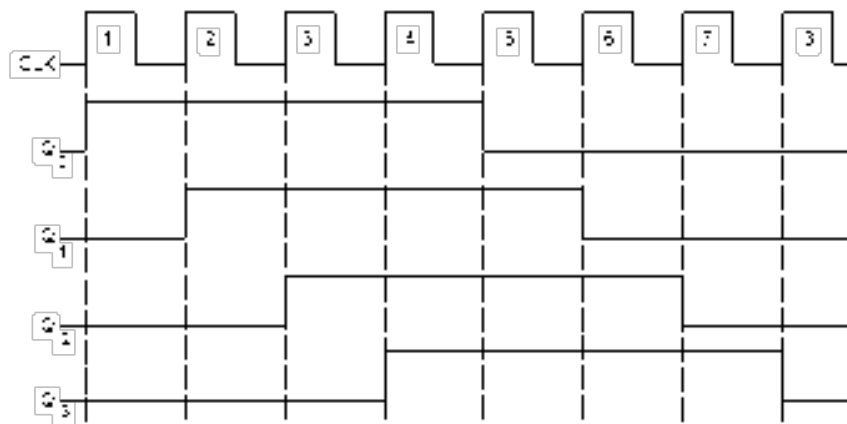


Fig.10 Timing Diagram of 4-bit Johnson Counter

Q.7 a. What is Magnitude Comparator? Explain 2-bit Magnitude Comparator with the help of truth table.

Answer:

Magnitude Comparator:

The comparison of two numbers is an operation that determines if one number is greater than, less than, or equal to the other number. A magnitude comparator is a combinational circuit that compares two numbers, A and B, and determines their relative magnitudes. The outcome of the comparison is specified by three binary variables that indicate whether $A > B$, $A = B$, or $A < B$.

Comparators can be designed for comparing multibit numbers. Figure 11 shows the block diagram of an n -bit comparator. It receives two n -bit numbers A and B as inputs and the outputs are $A > B$, $A = B$, and $A < B$. Depending upon the relative magnitude of the two numbers, one of the outputs will be HIGH. Table 7 gives the truth table of a 2-bit comparator.

(I) If the magnitude of the inputs A and B are equal (i.e., $A = B$):

Consider two numbers, A and B as inputs with two digits each i.e., A_1, A_0 and B_1, B_0 . The two numbers are equal if all pairs of significant digits are equal i.e., if $A_1 = 0, A_0 = 0, B_1 = 0, B_0 = 0$, then $A_1 = B_1$ and $A_0 = B_0$. For example if $A_1 = 0, A_0 = 0, B_1 = 0, B_0 = 0$, then pairs of significant digits i.e., $A_1 = B_1 = 0$ and $A_0 = B_0 = 0$. Output for this combination becomes 1 for $A = B$ and 0 for $A < B$ and $A > B$. This is given in the Truth Table.

(II) If the magnitude of the input A is greater than or less than B (i.e., $A > B$ or $A < B$):

To determine if A is greater than or less than B, we inspect the relative magnitude of pairs of significant digits starting from the most significant position. If the two digits are equal, we compare the next lower significant pair of digits. This comparison continues until a pair of unequal digits is reached.

(i) If the input A is greater than B (i.e., $A > B$): If the corresponding digit of A is 1 and that of B is 0, we conclude that $A > B$. For example if $A_1 = 0, A_0 = 1, B_1 = 0, B_0 = 0$, then pairs of significant digits are $A_1 = B_1 = 0$, and A_0 (i.e., digit 1) $>$ B_0 (i.e., digit 0). This is shown in the Truth Table.

(ii) If the input A is less than B (i.e., $A < B$):

If the corresponding digit of A is 0 and that of B is 1, we conclude that $A < B$. For example if $A_1 = 0, A_0 = 0, B_1 = 0, B_0 = 1$, then pairs of significant digits are $A_1 = B_1 = 0$, and A_0 (i.e., digit 0) $<$ B_0 (i.e., digit 1). This is shown in the Truth Table.

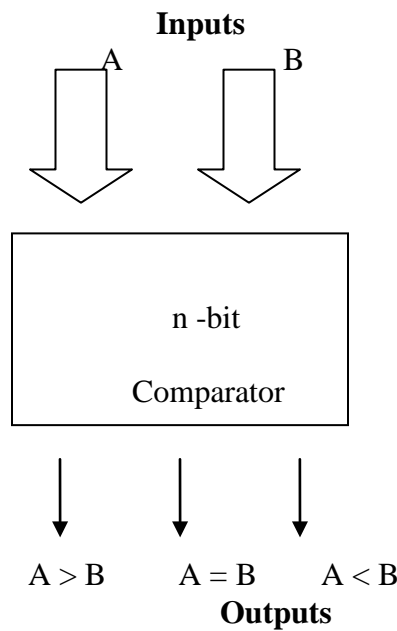


Fig.11 Block diagram of n-bit comparator

Inputs				Outputs		
A ₁	A ₀	B ₁	B ₀	A > B	A = B	A < B
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

Table 7 Truth Table of a 2-Bit Comparator

b. What is an Encoder? Draw the truth table for 10-line Decimal to 4-line BCD Encoder and implement the logic diagram from the truth table.

Answer:

Encoder: An Encoder is a combinational logic circuit which converts alphanumeric characters into Binary codes. It has 2n (or less) input lines and n output lines. An Encoder may be Decimal to Binary, Hexadecimal to Binary, and Octal to BCD etc.

.Decimal to BCD Encoder: This encoder has 10 inputs (for decimal numbers 0 to 9) and 4 outputs for the BCD number. Thus it is a 10 line to 4 line encoder. Table 8 lists the decimal digits and the equivalent BCD numbers. From the table, we can find the relationship between decimal digit and BCD bit. MSB of BCD bit is Y₃. For decimal digits 8 or 9, Y₃ = 1. Thus we can write OR expression for Y₃ bit as

$$Y_3 = 8 + 9$$

Similarly, Bit Y₂ is 1 for decimal digits 4, 5, 6 and 7. Thus we can write OR expression

$$Y_2 = 4 + 5 + 6 + 7$$

$$Y_1 = 2 + 3 + 6 + 7$$

$$Y_0 = 1 + 3 + 5 + 7 + 9$$

The logic circuit for the expressions (Y₀, Y₁, Y₂, Y₃) is shown in fig.12.

When a High appears on any of input lines the corresponding OR gates give the BCD output. For e.g., if decimal input is 8, High appears only on output 3 (and LOW on Y₀,

Y_1, Y_2), thus giving the BCD code for decimal 8 as 1000. Similarly, if decimal input is 7, then High appears on outputs Y_0, Y_1, Y_2 (and LOW on Y_3), thus giving BCD output as 0111.

Decimal Digit	BCD Code			
	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Table 8 The decimal digits and the equivalent BCD numbers

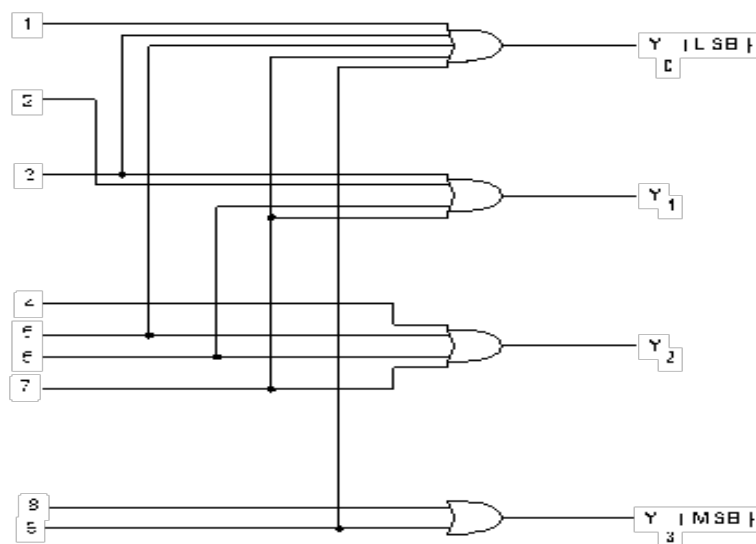


Fig. 12 Logic diagram for Decimal to BCD Encoder

Q.8 a. Design a Mod-6 Synchronous Counter and draw its designed logic diagram

Answer:

Design of Mod-6 Counter: The Mod-6 synchronous counter, have six counter states (i.e., from 0 to 6). The counter design table for this counter lists the three flip-flop and their states as 0 to 6 and the six inputs for the three flip-flops. The flip-flop inputs required to step up the counter from the present to the next state is worked out with the help of the

excitation table. The desired counter states and the $J K$ inputs required for counter flip-flops are given in the counter design table shown in Table No. 9.

Input pulse count	Counter States			Flip-Flop Inputs					
	A	B	C	J_A K_C	K_A	J_B	K_B	J_C	
0	0	0	0	1	X	0	X	0	X
1	1	0	0	X	1	1	X	0	X
2	0	1	0	1 X	X	X	0	0	0
3	1	1	0	X X	1	X	1	1	1
4	0	0	1	1 0	X	0	X	X	X
5	1	0	1	X	1	0	X	X	1
6(0)	0	0	0						

Table 9 Counter Design Table for Mod-6 Counter

Flip-Flop A:

The initial state is 0. It changes to 1 after the clock pulse. Therefore, J_A should be 1 and K_A may be 0 or 1 (that is X). In the next state 1 changes to 0 after the clock pulse. Therefore, J_A may be 0 or 1 (i.e., X) and K_A should be 1.

Flip-Flop B:

The initial state is 0 and it remains unchanged after the clock pulse. Therefore, J_B should be 0 and K_B may be 0 or 1 (that is X). In the next state 0 changes to 1 after the clock pulse. Therefore, J_B should be 1 and K_B may be 0 or 1 (i.e., X).

Flip-Flop C:

The initial state is 0 and it remains unchanged after the clock pulse. Therefore J_C should be 0 and K_C may be 0 or 1 (i.e., X). In the next state, it remains unchanged after the clock pulse. Therefore, J_C should be 0 and K_C may be 0 or 1 (i.e., X). The JK inputs required for this have been determined with the help of the excitation table, shown in Table No.8. The flip-flop input values are entered in Karnaugh maps shown in Fig. 13 ((i), (ii), (iii), (iv), (v), and (vi)) and a boolean expression is found for the inputs to the three flip-flops and then each expression is simplified. As all the counter states have not been utilized, Xs (don't) are entered to denote un-utilized states. The simplified expressions for each input have been shown under each map. Finally, these minimal expressions for the flip-flop inputs are used to draw a logic diagram for the counter shown in fig.14.

As before, the JK inputs required for this have been determined with the help of the excitation table, (Table 9.). These input values are entered in Karnaugh maps Fig. 13 [i to vi] and a boolean expression is found for the inputs to the three flip-flops and then each

expression is simplified. Xs have been entered in those counter states which have not been utilized. The simplified expressions for each input have been shown under each map and finally a logic diagram based on these expressions is drawn and is shown in Fig. 13 [i to vi].

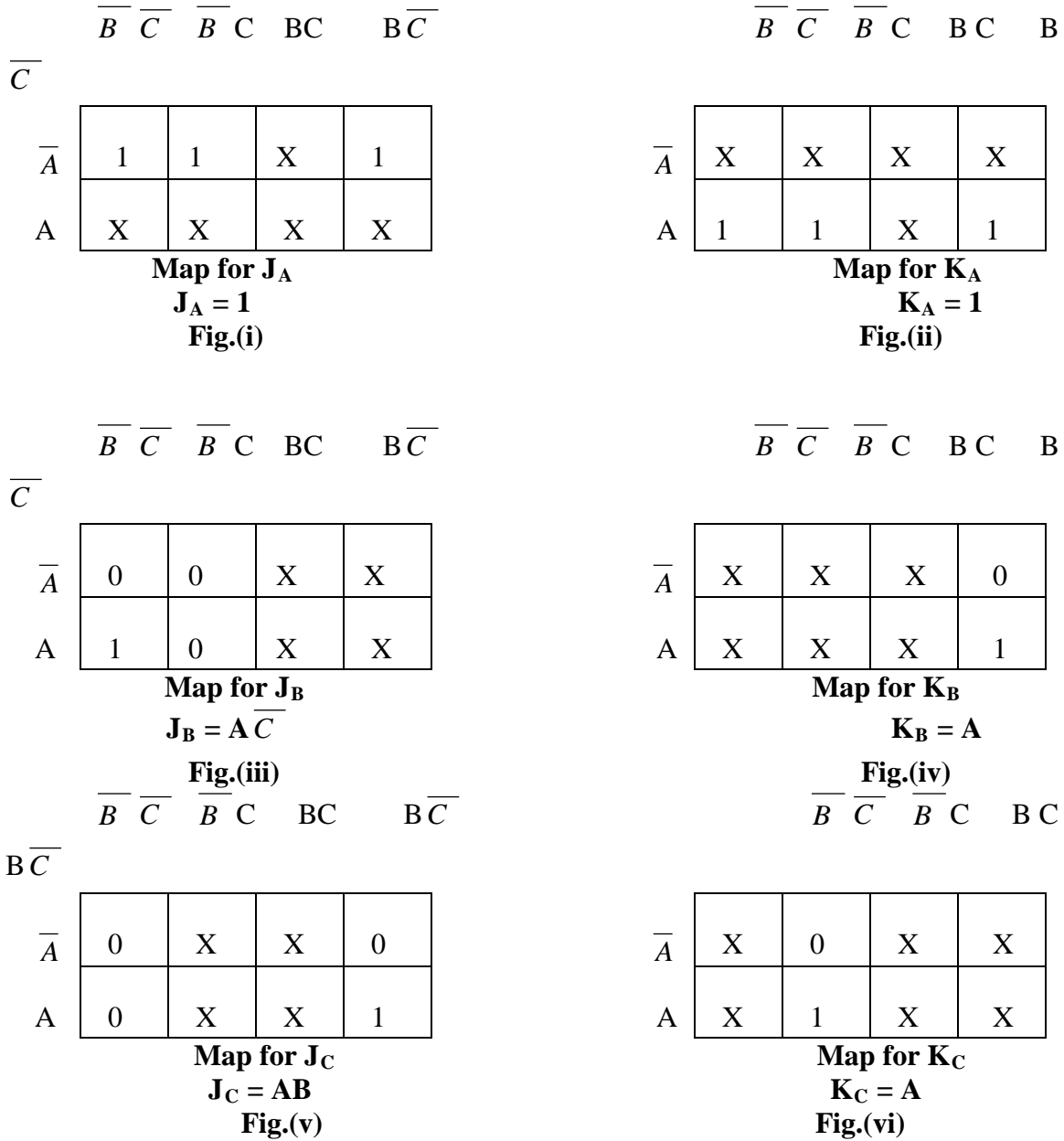


Fig. 13 Karnaugh Maps for $J_A, K_A, J_B, K_B, J_C, K_C$

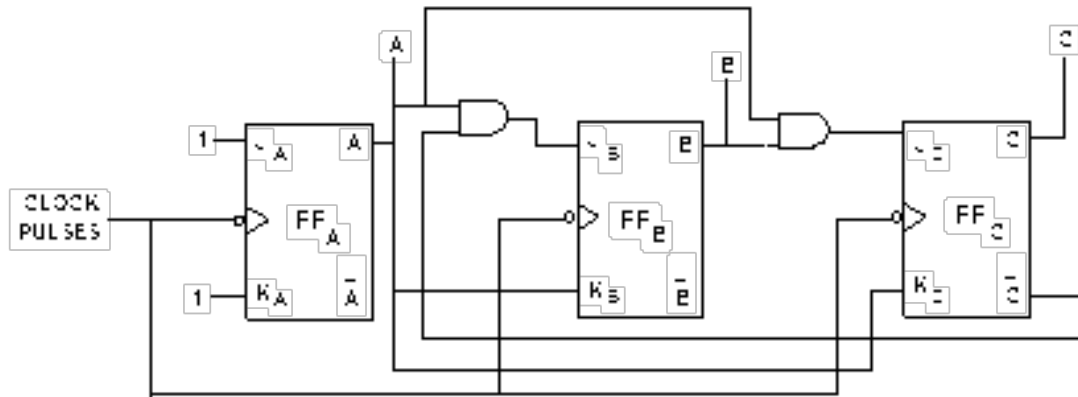


Fig. 14 Logic Diagram for MOD-6 Synchronous Counter

b. Draw the logic diagram for 4-bit Serial Input and Serial Output Shift Register and explain its working with timing waveform.

Answer:

SERIAL IN - SERIAL OUT SHIFT REGISTER:

Fig. 15 shows a 4 bit serial in - serial out shift register consisting of four D flip flops FF_0, FF_1, FF_2 and FF_3 . As shown it is a positive edge triggered device. The working of this register for the data 1010 is given in the following steps.

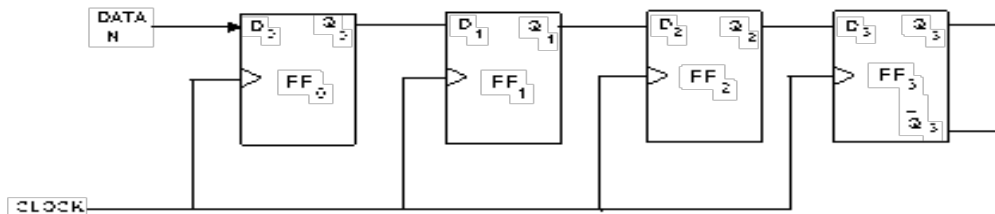


Fig. 15 Logic Diagram of 4-bit Serial In – Serial Out Shift Register

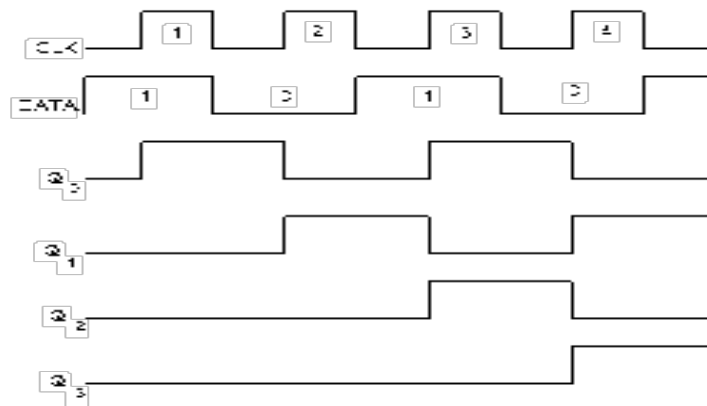


Fig.16 Output Waveforms of 4-bit Serial-in Serial-out Register

1. Bit 0 is entered into data input line. $D_0 = 0$, first clock pulse is applied, FF_0 is reset and stores 0.
2. Next bit 1 is entered. $Q_0 = 0$, since Q_0 is connected to D_1 , D_1 becomes 0.
3. Second clock pulse is applied, the 1 on the input line is shifted into FF_0 because FF_0 sets. The 0 which was stored in FF_0 is shifted into FF_1 .
4. Next bit 0 is entered and third clock pulse applied. 0 is entered into FF_0 , 1 stored in FF_0 is shifted to FF_1 and 0 stored in FF_1 is shifted to FF_2 .
5. Last bit 1 is entered and 4th clock pulse applied. 1 is entered into FF_0 , 0 stored in FF_0 is shifted to FF_1 , 1 stored in FF_1 is shifted to FF_2 and 0 stored in FF_2 is shifted to FF_3 . This completes the serial entry of 4 bit data into the register. Now the LSB 0 is on the output Q_3 .
6. Clock pulse 5 is applied. LSB 0 is shifted out. The next bit 1 appears on Q_3 output.
7. Clock pulse 6 is applied. The 1 on Q_3 is shifted out and 0 appears on Q_3 output.
8. Clock pulse 7 is applied. 0 on Q_3 is shifted out. Now 1 appears on Q_3 output.
9. Clock pulse 8 is applied. 1 on Q_3 is shifted out.
10. When the bits are being shifted out (on CLK pulse 5 to 8) more data bits can be entered in.

Fig. 16 shows the output waveforms of 4-bit Serial-in Serial-out Register.

Q.9 a. Draw and explain the architecture of 16×8 ROM.

Answer:

Architecture of 16×8 ROM:

A read-only memory is an array of selectively open and closed unidirectional contacts. A 16×8

ROM array is shown in Fig. 17. To select any one of the 16 bits, a 4-bit address (A_3, A_2, A_1, A_0) is required. The lower order two bits (A_1, A_0) are decoded by the decoder D_L which selects one of the four rows, whereas the higher order two bits (A_3, A_2) are decoded by the decoder D_H which activates one of the four column sense amplifiers.

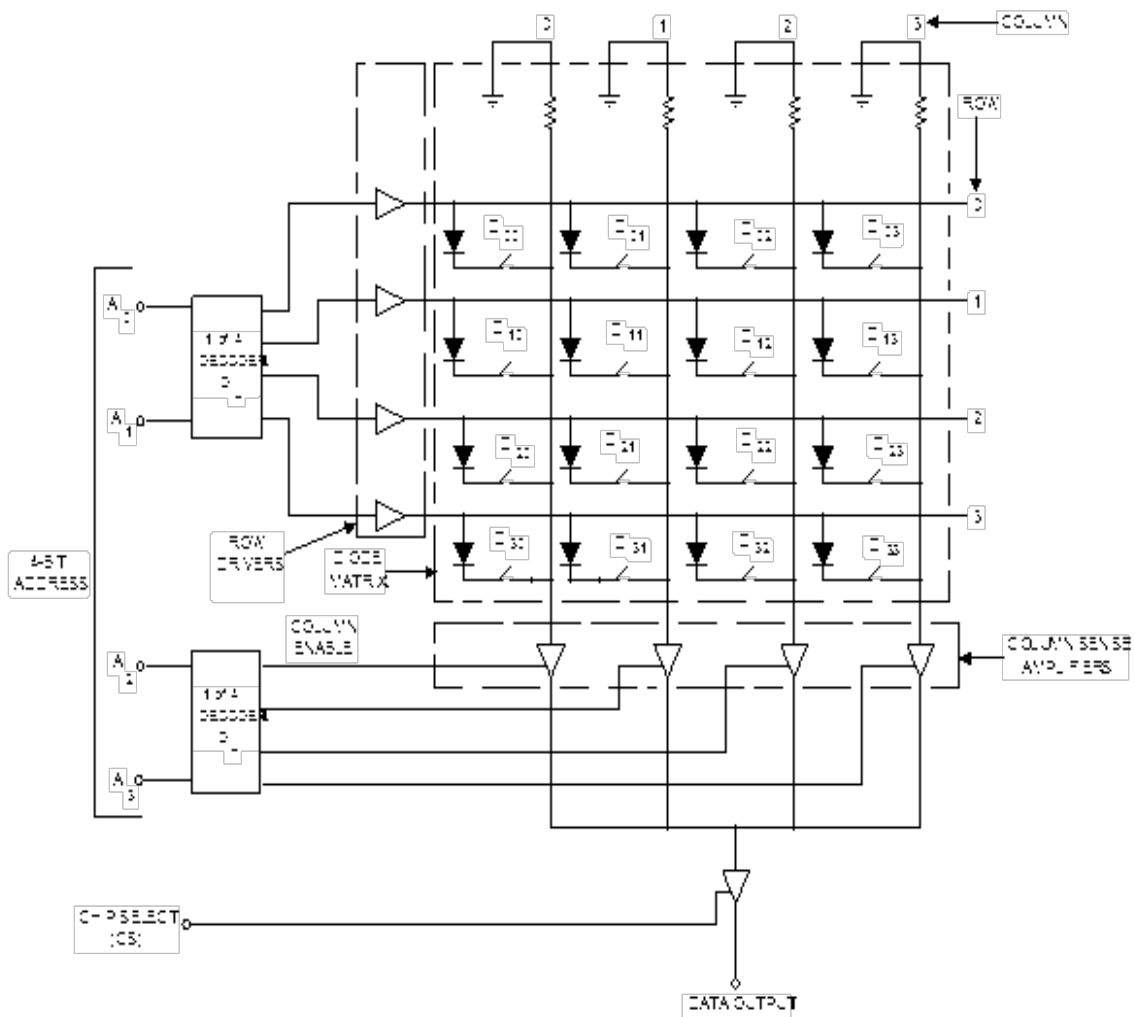


Fig. 17 Logic Diagram of 16-bit ROM array

The diode matrix is formed by connecting one diode along with a switch between each row and column. For example the diode D_{21} is connected between row 2 and column 1. The output is enabled by applying logic 1 at the chip select (CS) input. Programming a ROM means to selectively open and close the switches in series with the diodes. For example, if the switch of diode D_{21} is in closed position and if the address input is 0110, the row 2 is activated connecting it to the column 1. Also the sense amplifier of column 1 is enabled which gives logic 1 output if the chip is selected ($CS = 1$). This shows that a logic 1 is stored at the address 0110. On the other hand if the switch of diode D_{21} is open, logic 0 is stored at the address 0110.

b. Differentiate between Static RAM and Dynamic RAM.

Answer:

Differentiation between Static RAM and Dynamic RAM:

Static RAMs store ones and zeros using conventional FLIP-FLOPs. Whereas, the memory cells of dynamic RAMs are basically charge storage capacitors with driver transistors. The presence or absence of charge in a capacitor is interpreted as Logic 1 or 0.

Static RAMs do not require refreshing because there is no problem of charge leaking-off in FLIP-FLOPs whereas Dynamic RAMs require periodic charge refreshing to maintain data storage because the charge stored on capacitors leak-off with time.

Static RAMs are slower but easier to drive than dynamic memories, which generally require clock signals in addition to extra power supplies whereas dynamic circuits usually require externally generated clock voltages.

Advantages of Static RAMs over Dynamic RAMs:

- (i) Higher speed of operation (faster) i.e, lower access-time.
- (ii) Does not require refreshing.

Advantages of Static RAMs over Dynamic RAMs:

- (i) Higher number of bits storage on a given silicon chip area. i.e., Higher packaging density.
- (ii) Lower power consumption.

Text Book

**Digital Systems - Principles & Applications, Ronald J. Tocci, Neal S. Wildmer,
Gregory L. Moss, 9th Edition, Pearson Education, 2008**