



ZONE TECH
Best Institute For Assistant & Junior Engineer

RPSC-AEn Mains

Work Book Solution

for

Digital Electronics



Visit Our Website
www.zonetech.in



DOWNLOAD NOW

<https://zonetech.in/downloads>

PART - A

2 Marks Questions

1.
Sol. $9 \times 16^3 + 7 \times 16^2 + 4 \times 16^1 + 5 \times 16^0$
Hexadecimal Number- $(9745)_{16}$
convert into Binary- 1001 0111 0100 0101
Number of 1 is- 8
2.
Sol. $4 \rightarrow 0100$
 $F \rightarrow 1111$
 $A \rightarrow 1010$
 $D \rightarrow 1101$
Binary Equivalut $(0100\ 1111\ 1010\ 1101)_2$
3.
Sol. $0.6875 \times 2 = 1.375 \quad 1$
 $0.375 \times 2 = 0.750 \quad 0$
 $0.750 \times 2 = 1.500 \quad 1$
 $0.500 \times 2 = 1.000 \quad 1$
 $(0.6875)_{10} \rightarrow (0.1011)_2$
4.
Sol. Binary Representation of 17 = 10001
1's complement of 01110
2's complement = 01111
2's Complement of -17 = 01111
5.
Sol. ASCII is 7 bit code which mostly used in micro computers. This Code represent a character with seven bits, which can stored as one byte with one bit used.
6.
Sol. This code detects as well as corrects the error is an information The's Codes transmitts additional parity bits in addition to information bits. These parity will are used to delects and Correct the error.
7.
Sol. In gray code every New code differs from the previous code only by a single bit. That's why It is also called Reflacted Code. It can be used for measurement of shaft speed.
8.
Sol.
$$Y = \overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} + A\overline{B}\overline{C} + A\overline{B}C$$
$$= \overline{A}\overline{C}(\overline{B} + B) + A\overline{C}(\overline{B} + B)$$
$$= \overline{A}\overline{C} + A\overline{C}$$
$$= (\overline{A} + A)\overline{C}$$
$$= \overline{C}$$
9.
Sol. Range of 1's complement Number $= -(2^{n-1} - 1)$ to $(2^{n-1} - 1)$
 $n = 6$
 $= -(2^{6-1})$ to $(2^{6-1} - 1)$
 $= -(2^5 - 1)$ to $(2^5 - 1)$
 $= -31$ to $+31$



10.

Sol. Binary Number for 4 = (0100),

$$\begin{array}{cccc} & 0 \oplus & 1 \oplus & 0 \oplus & 0 \\ \text{Gray Code} & \downarrow & \downarrow & \downarrow & \downarrow \\ & 0 & 1 & 1 & 0 \end{array}$$

Gray code of Number 4 is given by 0110.

11.

Sol. $Y = x \oplus \bar{x}$

$$Y = x\bar{x} + \bar{x}x$$

$$Y = x + \bar{x}$$

$$Y = 1$$

$$[x + \bar{x} = 1]$$

12.

Sol. BCD Number 00010010

Decimal Number 1 2

Decimal Number is $(12)_{10}$.

13.

$$\begin{aligned} \text{Sol. } y &= \overline{\overline{AB} + \overline{CD} \cdot \overline{EF}} \\ &= \overline{\overline{AB} + \overline{CD} \cdot \overline{EF}} \\ &= AB + CD + EF \end{aligned}$$

14.

Sol. $(64)_{16} + (80)_{10}$

Convert Hexadecimal Number into decimal Number to

$$64 \rightarrow 6 \times 16^1 + 4 \times 16^0$$

$$96 + 4 = 100$$

$$\text{Now } (100)_{10} + (80)_{10} = (180)_{10}$$

15.

Sol. $= (\bar{x} + \bar{z})y$

16.

Sol. • Switch B and C are connected in parallel which Represent OR logic gate = $B + C$ 

• Now switch are connected in series Represent AND logic
 $y = A (B + C)$

17.

Sol. When each product or sum term of a boolean function contain all the logical variables of the function the representation of the function is called canonical form.

18.

Sol. If Each term in sop form contain all the variable of the function then each term is called minterm or Standard product and representation is called minterm form. Ex.- $\overline{A}\overline{B}\overline{C}\overline{D} + A\overline{B}CD + A\overline{B}C\overline{D}$

19.

Sol. If each term in pos form contain all the variables of the function then each terms is called maxterms or standard sum and function is called Maxterms form.

20.

$$\begin{aligned}
 \text{Sol. } y &= ABC\bar{C} + \bar{A}\bar{B}C + ABC \\
 &= AB\bar{C}\bar{C} + C + \bar{A}\bar{B}C \quad [C + \bar{C} = 1] \\
 &= AB + \bar{A}\bar{B}C
 \end{aligned}$$

21.

$$\begin{aligned}
 \text{Sol. } y &= \bar{A}\bar{B} + \bar{A}B + A\bar{B} \\
 &= \bar{A} + \bar{B} + \bar{A}B + A\bar{B} \\
 &= \bar{A} + \bar{A}B + \bar{B} + A\bar{B} \\
 &= \bar{A}[1+B] + \bar{B}[1+A] \\
 &= \bar{A} + \bar{B}
 \end{aligned}$$

22.

Sol. Combinational circuits are the circuits whose output depends only on present inputs. These circuits do not have any feed back or memory Element.

23.

Sol. It is a logic circuit that perform addition of two one bit Numbers. It has two input and two outputs. Outputs of half adder are sum and carry of Result.

24.

Sol. A full subtractor is a combinational circuit which performs subtraction of three bits called minuend (A_n) subtract end (B_n) and borrow from previous stage. It has two outputs called difference (D_n) and borrow (C_n) of Result.

25.

Sol. A decoder is a Combinational circuit that Converts binary information from 'n' input lines to a maximum of 2^n unique output lines. if n bit Combination has some don't care conditions then decoder decodes the binary informations at input lines.

$$\text{size of decoder} = n \times 2^n [m \leq 2^n]$$

where n = Number of input lines

2^n = Number of output lines

26.

Sol. Multiplexing means transmitting a large Number of information unit over a smaller number of channels or lines.

Multiplexers are digital Combinational circuits which select one line from 2^n input lines by using 'n' selection lines and send it to a single output line.

27.

Sol. An encoder is a digital Combinational circuit which produces reverse operation from that of a decoder. An encoder has 2^n input and n output lines.

28.

Sol. Demultiplexers is a Combinational circuit which receive information on a single line and send it to 2^n possible lines by using 'n' selection lines.

29.

Sol. Sequential circuit are those circuit whose present output depends on present input as well as previous output. These circuit have feedback from output of circuit. These circuit include memory elements.

30.

Sol. Flip-flop is a basic digital memory circuit or one bit memory cell. It acts as a bistable multivibrator. It has two stable state 0 and 1.

PART - B

5 Marks Questions

31.

Sol.

A \ BC	BC			
	00	01	1	10
0	1	1	1	1
1	1	1		1

$$Y = \bar{A} + \bar{B} + \bar{C}$$

32.

Sol.

$$\begin{aligned}
 Y &= ABC\bar{C} + A\bar{B}C + A\bar{B}\bar{C} \\
 &= ABC\bar{C} + A\bar{B}C + A\bar{B}\bar{C} \\
 &= A\bar{C}(B + \bar{B}) + A\bar{B}C \\
 &= A\bar{C} + A\bar{B}C \\
 &= A(\bar{C} + \bar{B}C) \\
 &= [(\bar{C} + \bar{B})(\bar{C} + C)]
 \end{aligned}$$

33.

Sol.

$$\begin{aligned}
 F &= \bar{S}_1\bar{S}_0I_0 + \bar{S}_1S_0I_1 + S_1\bar{S}_0I_2 + S_1S_0I_3 \\
 F &= \bar{x}\bar{y}L + \bar{x}y.o + x\bar{y}o + xy.1 \\
 F &= \bar{x}\bar{y} + xy
 \end{aligned}$$

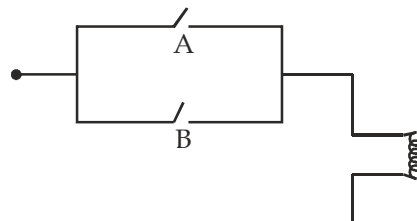
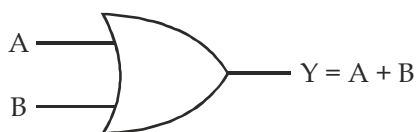
34.

Sol.

$$\begin{aligned}
 F &= A\bar{B}C\bar{D} + \bar{A}B\bar{C}\bar{D} + A\bar{B}C\bar{D} + \bar{A}B\bar{C}\bar{D} \\
 &= C\bar{D}[A\bar{B} + \bar{A}B] + CD[A\bar{B} + \bar{A}B] \\
 &= [C\bar{D} + CD][A\bar{B} + \bar{A}B] \\
 &= (A \oplus B)(C \oplus D)
 \end{aligned}$$

35.

Sol. OR Gate:- This operation is represented by (+)
 $y = A + B$



Truth Table

A	B	$Y=A+B$
0	0	0
0	1	1
1	0	1
1	1	1

Note:-

- (i) Output is high if either of input of two input OR gate is high.
- (ii) 1 is disable for OR gate.
- (iii) When one of the input OR gate is '0' then the output is same as another input.

36.

Sol.

Cominational Circuit	Seprential Circuit
• Present output depends only on present input	• Present output depends on both present input as well as past output
• Memory less system	• Memory system
• Faster	• Slower
• Easy to design	• Harder to design
• No feedback present	• Feedback present
• No clock pulse required	• Clock pulse required

37.

Sol.

Synchronous counter	Asynchronous counter
1. Same clock pulse is applied to individual flip flop	1. Clock signal is applied only the first flip flop
2. Any sequence can be generate	2. Fixed sequence
3. No Decoding error	3. Secoding error exist
4. Design is complex as number of bits increases Ex.- Ring counter, Johnson counter	4. Design easy, even more bits number of Ex.- Ripple up or Ripple down counter

38.

Sol.

Latch	Flip-Flop
1. Latch use level triggering	1. Flip-Flope use edge triggering
2. No clock pulse	2. Clock pulse
3. Build from gates	3. Build from latch
4. The output changes as per the input till enable is high	4. The output changes as per the input only at triggering point
5. Buffer	5. 1 bit memory element
6. output can not predict	6. output can predict

39.

Sol.

Static RAM	Dynamic RAM
1. Data is store like Flip-Flop	1. Data is stored in MOS capacitor
2. BJT, MOSFET are used	2. MOSFET is used
3. Faster then DRAM	3. Slower than SRAM
4. Power dissipation is more	4. Power dissipation in low
5. Density is Low	5. Density is high
6. Used as cache memory	6. Used as main memory
7. No refreshing is required	7. Refreshing is required

40.

Sol.

- A systematic & simple way of minimization of Boolean Algebra.
 - Gray code is used in k-map.
 - Number of cells = 2^N , N = Number of variables.
 - Don't care → Considered when this is helping in minimization of Boolean Algebra.
- 3 variables k-map

A \ BC	BC			
	00	01	11	10
0	0	1	3	2
1	4	5	7	6

41.

Sol.

RAM	ROM
1. It is read/write memory	1. It is read only memory
2. Random access momory	2. Read only memory
3. used to store the data that has to be currently processed by CPU	3. It stores the instructions required during boot strap of the computer.
4. It is volatile memory	4. It is non volatile memory

Design two bit comprator?

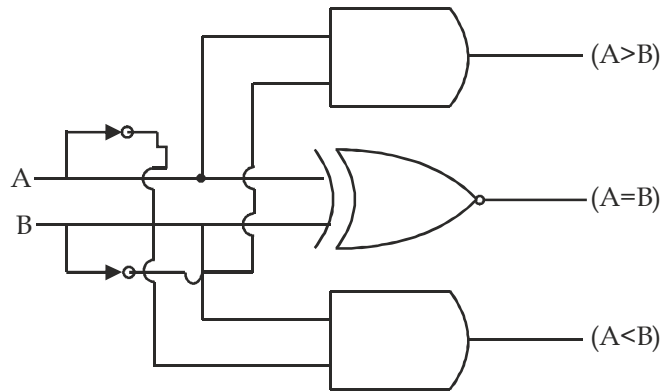
A	B	A < B	A = B	A > B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

$$(A < B) = \overline{A}B$$

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

$$(A > B) = A\overline{B}$$





PART - C

20 Marks Questions

42.

Sol. **Binary Codes For The Decimal Digits-** Binary codes can be classified as numeric codes and alphanumeric codes. Alphanumeric codes represent the alphabet and decimal numbers as a sequence of 0s and 1s. ASCII code is an example of alphanumeric code which represents alphabet and decimal numbers. Numeric codes represent decimal digits and are called Binary Coded Decimal (BCD) codes.

1. **Binary Coded Decimal (BCD) Code-** This is a four bit code and also it is sometimes called 8-4-2-1 code. This expresses each decimal digit by its four binary equivalent. For example

326 is written as $\begin{matrix} 3 & 2 & 6 \\ \downarrow & \downarrow & \downarrow \\ 0011 & 0010 & 0110 \end{matrix}$ in BCD code. So 326 is

equivalent to 0011 0010 0110 in BCD Code. This is called 8-4-2-1 code because the weights in the four bit group reading from left to right are 8, 4, 2 and 1 same as that of binary numbers. The advantage of this system is that we need not remember the binary numbers for 0 to 9 as we encode only one decimal digit at a time.

BCD numbers are very useful for input and output operations in digital circuits. It is used to represent decimal digits in systems like pocket calculators, digital clocks, electronic counters, digital voltmeters etc.

2. **Excess-3 Code-** The Excess-3 code, another important BCD code, is an unweighted code. The Excess-3 code is particularly significant for arithmetic operations. When the sum of two decimal digits exceeds 9, they cannot be added using 8421 BCD code. Excess-3 code has no such limitation and considerably simplifies arithmetic operations.

A given decimal number can be converted into its Excess- 3 equivalent as follows:

1. Add 3 to each decimal digit in the given decimal number.
2. Replace each digit of the newly formed decimal number by its 4-bit binary equivalent.

It is clear that group 0000, 0001, 0010, 1101 and 1111 are not used in this code.

3. **Gray Code-** It is an unweighted code and not suitable for arithmetic operations but is quite useful for peripheral equipment, A/D counter etc. Each number in Gray code differs from its preceding number by single bit. For conversion of binary numbers to Gray Code the following rules are to be followed. The first Gray digit is the same as the first binary digit. Add the first two bits in mod-2 arithmetic, the sum is our next Gray digit. Repeat it until entire Gray code number is obtained.

Following steps should be followed for converting a given binary number to its Gray code equivalent:

1. Begin with the most significant bit (MSB). The MSB of the Gray code equivalent is the same as the MSB of given binary number.
2. The second most significant bit (adjacent to MSB) in the Gray code number is obtained by adding the MSB and the second MSB of the binary number and ignoring carry if any.
3. The third most significant bit in the Gray code number is obtained by adding the second MSB and third MSB in the binary number and ignoring carry if any.

This process continues till we obtain the LSB.

Alphanumeric Codes- In order to communicate, we need not only numbers but also letters of alphabet, punctuation marks and other special characters. These codes are called alphanumeric codes. Most of these codes, however, also represent symbols and various instructions necessary for conveying intelligible information.

Different codes are used for representing alphanumeric data stored in paper tape, punched card or magnetic tape for use by a line printer or typewriter. The most common alphanumeric codes are ASCII (American Standard Code for Information Interchange) and EBCDIC (Extended Binary Coded Decimal Interchange Code).

ASCII, pronounced "askee" is perhaps the most widely used alphanumeric code. It is a seven-bit code in which the decimal digits are represented by the 8-4-2-1 BCD code preceded by 011. It has 2 i.e., 128 possible code groups which is more than enough to represent all of the standard keyboard characters as well as control functions such as the (RETURN) and (LINEFEED). This code is used for the transfer of alphanumeric information between a computer and input/output devices such as video terminals or printers. A computer also uses it internally to store the information that an operator types in at the computer's keyboard.

EBCDIC is an eight-bit code in which the decimal digits are represented by the 8421 BCD code preceded by 111.

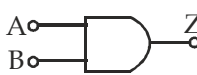
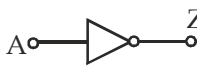
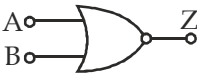
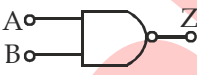
Self-Checking/Correcting Codes- A parity bit is introduced in the side of a code to detect the error. It can follow either odd parity code or even parity code.

We have an EVEN PARITY when an extra bit is added to a code so that the total number of 1s becomes an even number (this added bit could be a '0' or a '1') and an ODD PARITY where an extra bit is added to a code to make number of 1s as odd (this added bit could be a '0' or a '1'). For instance, if we have to add an even parity bit to 4-bit number 1101, it will be a '1' and if we have to add an odd parity bit to the same number, it would be a '0'. The parity check can be made at different points to look for any possible single bit error. One can appreciate that for a given code, the even parity bit is complement of the odd parity bit. We will also appreciate that addition of a single parity bit (evens or odd) cannot be used to detect 2-bit errors. It may, however, be mentioned that in normal operation, chances of 2-bit errors are very low. Nevertheless, there are some data storage media magnetic tapes for instance, that are highly prone to 2-bit errors

43.

Sol. LOGIC GATES-

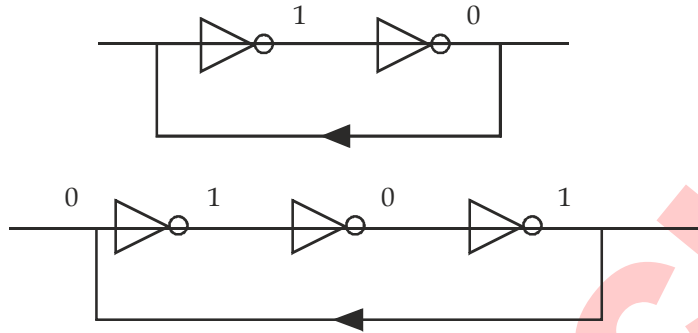
Boolean functions can be performed by digital circuits. Circuits. Boolean that perform complicated boolean functions can be subdivided into simpler circuits that perform simpler boolean functions. The circuits that perform the simplest boolean functions are taken as basic elements, called logic gates and are represented by specialised symbols. The logic gates are identified by the adjective representing the operation they perform. The most common logic gates are the OR, AND, NOT, NAND, NOR, Exclusive OR and Exclusive NOR gates. The NAND and NOR gates are called the universal gates. NOT gate or inverter is the only nontrivial single input gate.

Logic With IC	Symbol	Function	Truth Table	Remarks																		
OR (IC 7432)		$Z=A \text{ OR } B$ $=A+B$	<table><tr><th colspan="2">Inputs</th><th>Output</th></tr><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	Inputs		Output	A	B	Z	0	0	0	0	1	1	1	0	1	1	1	1	An OR gate is any or all gate. It has two or more inputs but only one output. In an OR gate if any one or more inputs are high, the output is high. If all the inputs are low, the output is low.
Inputs		Output																				
A	B	Z																				
0	0	0																				
0	1	1																				
1	0	1																				
1	1	1																				
AND (IC 7408)		$Z=A \text{ AND } B$ $=A.B$	<table><tr><th colspan="2">Inputs</th><th>Output</th></tr><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	Inputs		Output	A	B	Z	0	0	0	0	1	0	1	0	0	1	1	1	An AND gate is an all or nothing gate. It has two or more inputs but only one output. All inputs must be high to provide a high input.
Inputs		Output																				
A	B	Z																				
0	0	0																				
0	1	0																				
1	0	0																				
1	1	1																				
NOT (IC 7404)		$Z=\text{NOT } A$ $=\bar{A}$	<table><tr><th>Inputs</th><th>Output</th></tr><tr><th>A</th><th>Z</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	Inputs	Output	A	Z	0	1	1	0	A NOT gate (or inverter gate) has only one input and one output. It complements the input i.e., its output is always the opposite of the input.										
Inputs	Output																					
A	Z																					
0	1																					
1	0																					
NOR (IC 7402)		$Z=A \text{ NOT OR } B$ $=A \text{ NOR } B$ $=\overline{A+B}$	<table><tr><th colspan="2">Inputs</th><th>Output</th></tr><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	Inputs		Output	A	B	Z	0	0	1	0	1	0	1	0	0	1	1	0	A NOR gate is effectively an OR gate followed by an inverter. It has two or more inputs and one output. All inputs must be low to provide a high input. It acts as a negative AND gate.
Inputs		Output																				
A	B	Z																				
0	0	1																				
0	1	0																				
1	0	0																				
1	1	0																				
NAND (IC 7400)		$Z=A \text{ NOT AND } B$ $=A \text{ NAND } B$ $=\overline{A.B}$	<table><tr><th colspan="2">Inputs</th><th>Output</th></tr><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	Inputs		Output	A	B	Z	0	0	1	0	1	1	1	0	1	1	1	0	A NAND gate is effectively an AND gate followed by an inverter. It has two or more inputs and one output. All inputs must be high to get a low output. It acts as a negative OR gate.
Inputs		Output																				
A	B	Z																				
0	0	1																				
0	1	1																				
1	0	1																				
1	1	0																				
EX-OR (IC 7486)		$Z=A \text{ EX-OR } B$ $=A \oplus B$ $=A\bar{B} + \bar{A}B$	<table><tr><th colspan="2">Inputs</th><th>Output</th></tr><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	Inputs		Output	A	B	Z	0	0	0	0	1	1	1	0	1	1	1	0	An EX-OR gate is an anticoincidence gate. It is an inequality detector. It has two inputs and one output. It provides high output, only when its two inputs are different.
Inputs		Output																				
A	B	Z																				
0	0	0																				
0	1	1																				
1	0	1																				
1	1	0																				
EX-NOR (IC 74266)		$Z=\overline{A \text{ EX-OR } B}$ $=A \text{ EX-NOR } B$ $=\overline{A \oplus B}$ $=AB + \bar{A}\bar{B}$	<table><tr><th colspan="2">Inputs</th><th>Output</th></tr><tr><th>A</th><th>B</th><th>Z</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	Inputs		Output	A	B	Z	0	0	1	0	1	0	1	0	0	1	1	1	An EX-NOR gate is a coincidence gate. It has two inputs and one output. It produces high output, only when its two inputs are the same.
Inputs		Output																				
A	B	Z																				
0	0	1																				
0	1	0																				
1	0	0																				
1	1	1																				

Boolean algebra allows any combinational logic circuit to be constructed with AND, OR and NOT gates. Any combinational logic circuit may also be constructed solely with NAND gates, as well as solely with NOR gates.

Table provides the IC no, symbol, Boolean expression and Truth table of various logic gates.

When even number of NOT Gates are connected in feedback, then it acts like a bistable multivibrator (BMV).



When odd number of NOT Gates are connected in feedback, then it acts as an astable multivibrator (AMV) or square-wave generator or clock generator or ring oscillator.

All inverters take sometime to get response "Z", this time is termed the propagation delay time (t_{pd}).

Time period of AMV is given as

$$T = 2nt_{pd}$$

where n is the number of NOT Gates (or inverters), t_{pd} is propagation delay time of each inverter.

Frequency of AMV is given as

$$f = \frac{1}{T} = \frac{1}{2nt_{pd}}$$

44.

Sol. BOOLEAN ALGEBRA LAWS AND RULES

Boolean algebra may be defined with a set of elements, a set of operators and a number of unproved axioms or postulates. A set of elements is any collection of objects having a common property.

In Boolean algebra, use is made of binary digits to represent the two levels that occurs within digital logic circuits. A binary represents a high level while a binary 0 represents a low level in Boolean expressions and in terms of voltages there are two types of logic systems viz. positive level logic system and negative logic system.

A truth table lists all possible combinations of inputs and the corresponding outputs.

Boolean algebra is the foundation of logic circuits.

Commutative, associative and distributive laws are valid in Boolean algebra. Postulates of Boolean algebra are a set of logical expressions that we accept without proof and upon which we can build a set of useful theorems.

1. **Commutative Laws-** $A + B = B + A$ and $A \cdot B = B \cdot A$

2. **Associative Laws-** $A + (B + C) = (A + B) + C$
and $A \cdot (B \cdot C) = (A \cdot B) \cdot C$

3. **Distributive Laws-** $A \cdot (B + C) = A \cdot B + A \cdot C$

Boolean algebra rules are given below:

OR Rules $A + 0 = A$; $A + 1 = 1$; $A + A = A$; $A + \bar{A} = 1$

AND Rules $A \cdot 0 = 0$; $A \cdot 1 = A$; $A \cdot A = A$; $A \cdot \bar{A} = 0$

Complementation Rule $\bar{\bar{A}} = A$

Absorptive Rule $A + AB = A$; $A + \bar{A}B = A + B$;

$$(A + B)(A + C) = A + BC$$

BOOLEAN POSTULATES

There are five basic postulates of Boolean algebra. A list of these postulates is given below:

- | | | | |
|------|---------------------------------|------|---------------------------------|
| 1(a) | If $A = 1$, then $\bar{A} = 0$ | 1(b) | If $A = 0$, then $\bar{A} = 1$ |
| 2(a) | $0.0 = 0$ | 2(b) | $1 + 1 = 1$ |
| 3(a) | $1.1 = 1$ | 3(b) | $0 + 0 = 0$ |
| 4(a) | $1.0 = 0$ | 4(b) | $0 + 1 = 1$ |
| 5(a) | $\bar{1} = 0$ | 5(b) | $\bar{0} = 1$ |

Here we see that each postulate has two parts, a and b. They are called duals of each other. By dual we mean that 'b' form is made from the 'a' form by changing the Boolean operation AND to OR and logic 0 to 1 or 1 to 0.

DE MORGAN'S THEOREM

Two of the most important theorems of Boolean algebra are a contribution of a great mathematician name De Morgan. These theorems, called the De Morgan's theorems, are extremely useful in simplification of expressions in which a sum or product of variables is inverted. These two theorems are:

$$\overline{A+B} = \bar{A} \cdot \bar{B}$$

$$\overline{A \cdot B} = \bar{A} + \bar{B}$$

The theorem expressed by Eq. is stated as below:

The complement of the sum of two or more variables is equal to the product of the complements of the variables i.e., the complement of two or more variables ORed is the same as the AND of the complements of each individual variable. The theorem expressed by Eq. (20.2) is stated as below:

The complement of the product of two or more variables is equal to the sum of the complements of the variables i.e., the complement of two or more variables ANDed is the same as the OR of the complements of each individual variable.

DUALITY THEOREM

The duality theorem is one of those elegant theorems proved in advanced mathematics. Basically duality underlies all Boolean algebra. Each Boolean expression has its dual which is as true as the original expression. The dual of a given Boolean expression can be obtained by following the procedure given below:

1. Change each OR (+) sign to an AND ($\cdot$) sign.
2. Change each AND ($\cdot$) sign to an OR (+) sign.
3. Complement any 0 or 1 appearing in the expression.

45.

Sol. Two standard forms of logic expressions are sum of products (SOP) form and product of sum (POS) form. Any one of them may give a circuit with lesser hardware requirement. SOP can be written in terms of minterms while POS can be written in terms of maxterms.

A sum-of-product (SOP) expression is two and more AND functions ORed together. For instance $AB + \bar{B}\bar{C} + CD$ is SOP expression. An SOP form can also have a term with a single variable such as $A + \bar{A}\bar{B} + \bar{B}C$.

Product-of-Sums (POS) Form- The product-of-sums (POS) form can be thought of as the dual of the sum-of-products (SOP). It is, in terms of logic functions, the AND function of two or more ORed functions. For instance $(A + B)(C + D)$ is a product-of-sums expression. Several other examples of POS are:

$$(A + B)(\bar{A} + \bar{B}); (A + \bar{B})(A + B + C)(B + C); (A + B)(B + C + D); (A + B + C)(D + E + F + G)(A + E + G)$$

A product-of-sum expression can also contain a single variable term, such as

$$A(\bar{B} + \bar{C})(B + C)$$

Canonical Forms of Logical Expressions- When each term of a logic expression contains all literals, it is called a canonical form of the logical variables. If each term in SOP and POS forms contains all the literals then these are known as canonical or standard SOP and POS forms respectively. Each individual term in standard SOP form is called as minterm and in standard POS form as maxterm. SOP form can be converted to canonical SOP form by ANDing the terms in the expression with terms formed by ORing the variable and its complement which are not present in that term. Similarly, POS form can be converted to canonical POS form by ORing the terms in the expression with terms formed by ANDing variable and its complement which are not present in that term.

For instance $A + A\bar{B}$ can be converted into standard SOP form by ANDing the term A with terms $(B + \bar{B})$; formed by ORing the variable B and its complement $\bar{B}$ because B is not present in the first term.

Complementary Nature of Maxterms and Minterms- Since, the SOP and POS forms of Boolean expressions are complementary to each other, there is a complementary type of relationship between a function expressed in terms of minterms and in terms of maxterms.

For instance, if the minterm designation for a three variable expression is

$$Z = \sum m(3, 4, 6, 7),$$

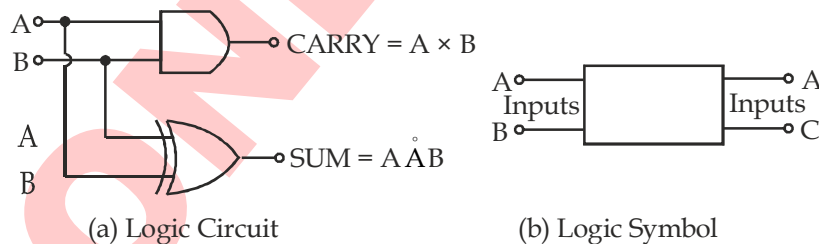
then the corresponding maxterm form is

$$Z = \prod M(0, 1, 2, 5)$$

Since, for a three-variable function, the total number of minterms and maxterms are $2^3 = 8$ and the corresponding numbers are 0 through 7. Out of this the terms corresponding to decimal numbers 3, 4, 6, 7 are minterms, and the terms corresponding to decimal numbers 0, 1, 2, and 5 are maxterms. Hence, if a logical function is specified in terms of minterm/maxterm, its maxterm/minterm representation can be determined by using this complementary property.

46.

Sol. Half Adder- A half-adder is an arithmetic circuit that adds two binary digits. It uses EX-OR gate and AND gate. Its outputs are SUM and CARRY. The circuit of a half adder is shown in Fig. Truth table of half adder is given in Table.



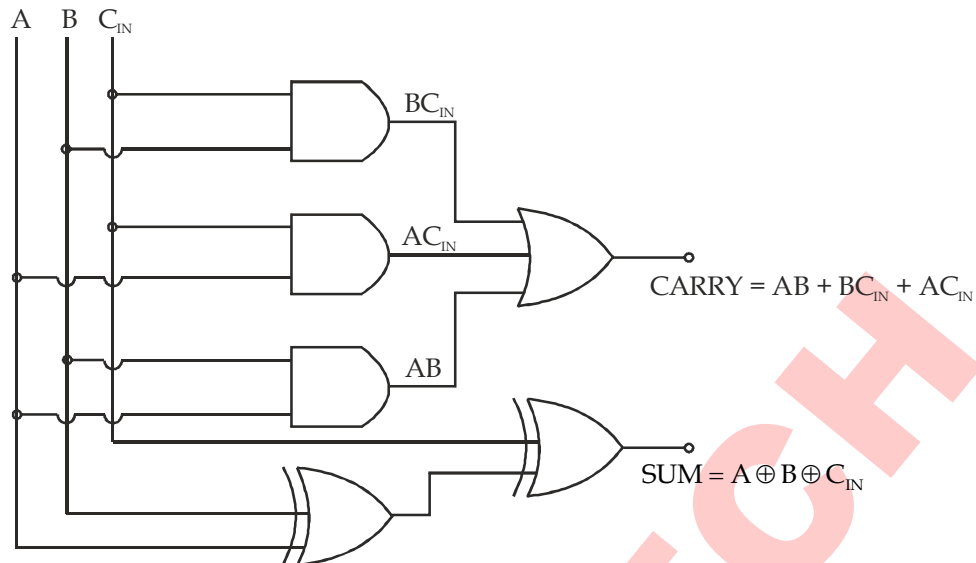
Truth Table For Half-Adder

Inputs		Outputs	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	1

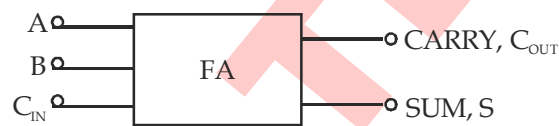
An half adder can add only two one bit numbers. It cannot accept a carry from a previous addition, Total number of NAND/ NOR gates required to implement an half adder is 5.

Full Adder- A full-adder can accept three inputs viz. the two bits of the two numbers and a CARRY IN from the lower order adder. Its outputs are SUM and CARRY. A full-adder circuit has two EX-OR gates, three AND gates and one OR gate.

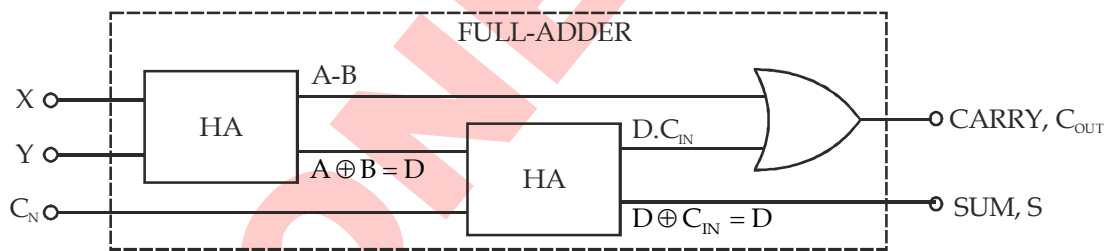




(a) Logic Circuit



(b) Logic Symbol



(c) Full-Adder Using Two Half-Adders

Inputs			Outputs	
A	B	C_{IN}	S	C_{OUT}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

A simple circuit and logic symbol of a full adder are shown in Fig. 20.8(a) and 20.8(b) respectively. The full-adder can do more than a million additions per second. Besides that, it never gets tired or bored or asks for a rest.

47.

Sol. Half Subtractor- A half-subtractor is an arithmetic circuit that subtracts one binary digit from another. It has one EX-OR gate, one NOT gate and one AND gate.

Logic symbol and logic circuit are shown in Figs. 20.9 (a) and 20.9(b) respectively. Truth table is given in Table 20.8.

Logical equations for the difference D and borrow B are given below:

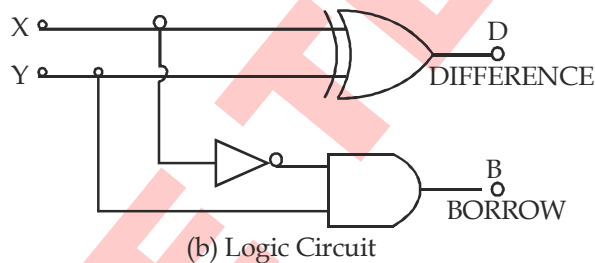
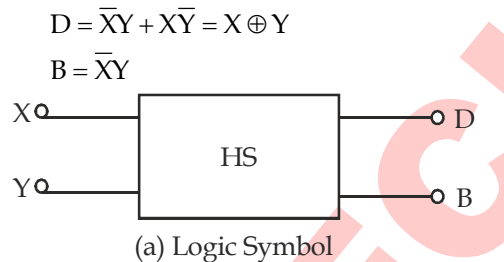


Fig.:- Half-Subtractor

Table: Truth Table For Half-Subtractor

Inputs		Outputs	
X	Y	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Total number of NAND/NOR gates required to implement H.S. is 5.

Full Subtractor- Just like a full-adder, a full-subtractor circuit is required to perform multibit subtraction wherein a borrow from the previous bit position may also be there. It has 3 inputs X (minuend), Y (subtrahend) and BIN (borrow from the previous stage) and two outputs D (difference) and BOUT (borrow), as shown in block diagram [Fig. 20.10 (a)]. Full-subtractor can be formed by using two half subtractors and one OR gate, The circuit is shown in Fig. (b).

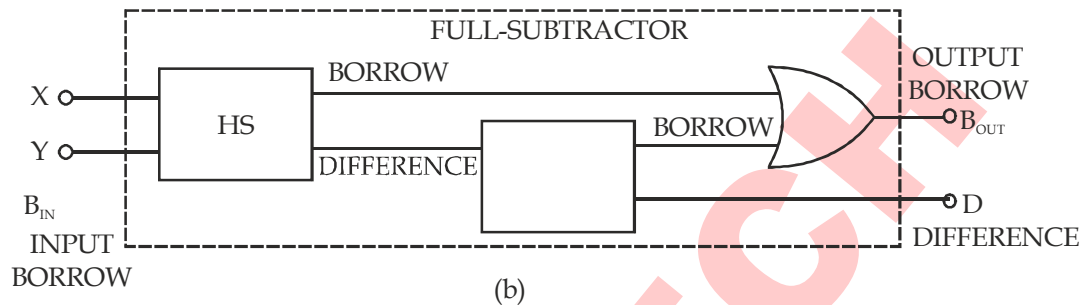
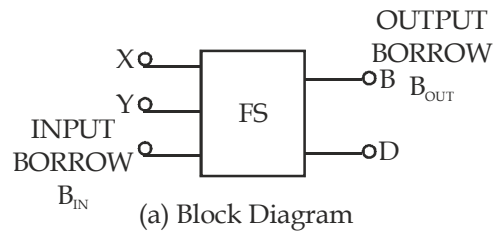


Fig.- Full-Subtractor

Note- For subtraction of n-bit numbers directly, we will have to cascade n full-subtractors. Truth table for full-subtractor is given in Table.

Table- Truth Table For Full-Subtractor

Inputs			Outputs	
X	Y	B_{IN}	D	B_{OUT}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Total number of NAND/NOR gates required to implement FS is 9.

48.

Sol. **MULTIPLEXER**

A multiplexer is a logic circuit that accepts several data inputs and allows only one of them at a time to get through to the output. Multiplexers are used for data selection, data routing, operation sequencing, waveform generation, logic function generation etc. Multiplexers are available as ICs. The standard packages are 2 : 1, 4 : 1, 8 : 1 and 16 : 1.

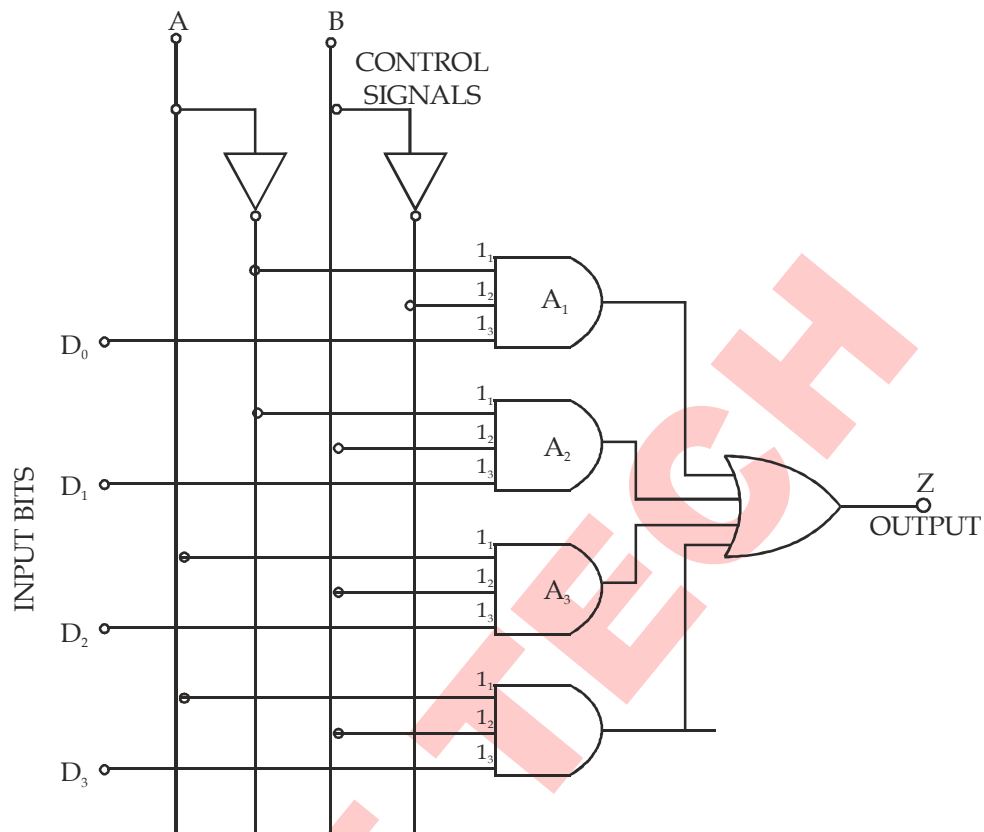


Fig.:- Four-To-One Multiplexer

Table- Truth Table For Four-To-One MUX

Data Select Inputs		Input Selected	Input	Output
A	B	D	D	Z
0	0	D_0	0 1	0 1
0	1	D_1	0 1	0 1
1	0	D_2	0 1	0 1
1	1	D_3	0 1	0 1

There are 2^n input lines where n is the select line. The selection of a particular input line is controlled by a set of select lines.

A circuit diagram for four-to-one multiplexer is given in Fig. Truth table for four-to-one multiplexer is given in Table.

The size of the MUX is specified by the 2^n of its input line and the single output line. MUX contains AND gate followed by an OR gate.



ENCODERS

An encoder converts an active input signal into a coded output signal. It has n input lines and m output lines. At a time only one of the input lines is active. Internal logic within the encoder converts this active input to a coded binary output with m bits.

One of the most commonly used input device for a digital system is a set of ten switches, one for each numeral. These switches generate 1 or 0 logic levels in response to turning them off or on. When a particular number is to be fed to the digital circuit in BCD code, the switch corresponding to that number is pressed. There is an IC available to perform this function (74147) which is a priority encoder. Circuit diagram of a decimal-to-BCD encoder is shown in Fig.

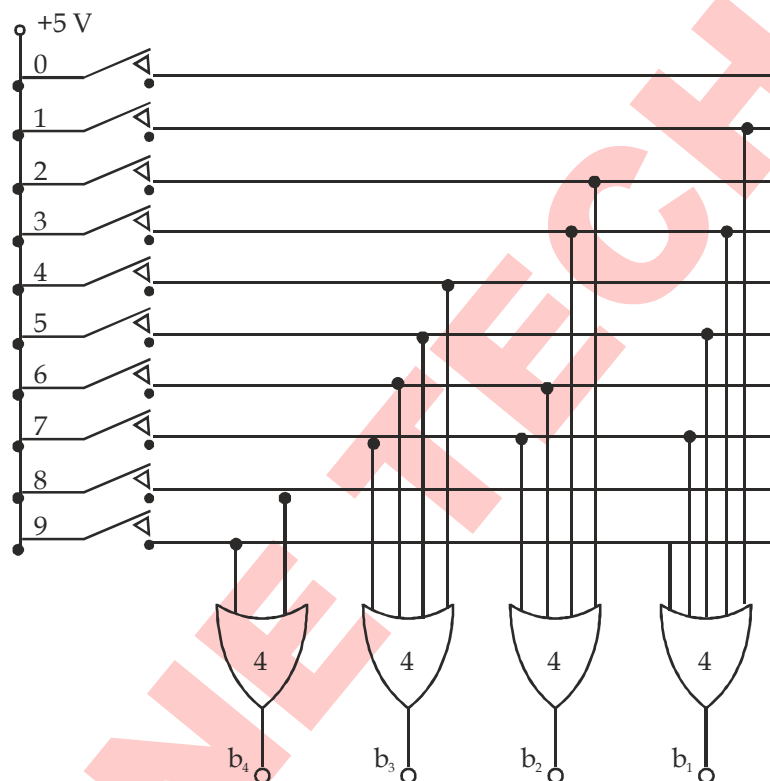


Fig.:- Decimal-To-BCD Encoder

TABLE- Truth Table For Decimal-To-BCD Encoder

Input										Output			
0	1	2	3	4	5	6	7	8	9	b_4	b_3	b_2	b_1
1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0	0	0	1	0	1	0
0	0	0	0	0	0	1	0	0	0	1	1	0	0
0	0	0	0	0	0	0	1	0	0	1	1	1	0
0	0	0	0	0	0	0	0	1	0	1	0	0	0
0	0	0	0	0	0	0	0	0	1	1	0	0	1

When switch 0 is pressed, all OR gates are inactive and so output by $b_4 b_3 b_2 b_1$, is 0000 which is a BCD equivalent of decimal digit 0.

When switch 1 is pressed, only OR gate 1 is active and so output $b_4 b_3 b_2 b_1$ is 0001 which is a BCD equivalent of decimal digit 1.

In the same way when switch 7 is pressed, OR gates 3, 2 and 1 are active, so output $b_4 b_3 b_2 b_1$ is 0111 which is again BCD equivalent of decimal digit 7.

Table is a truth table for decimal-to-BCD encoder.

49.

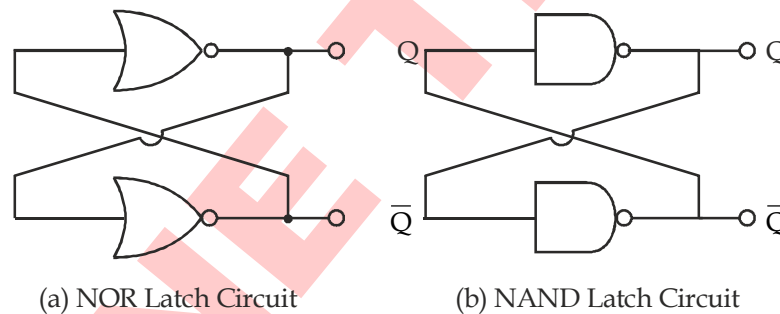
Sol. FLIP-FLOPS

A flip-flop (FF) is the basic memory element; it can store a 0 or a 1. A flip-flop is known more formally as a bistable multivibrator. It has two stable states. A FF can have one or more inputs. These inputs cause FF to switch back and forth (flip-flop) between its possible output states. FFs are used for data storage counting, frequency division etc.

1. Latches- Latches are bistable elements whose state depends on asynchronous inputs. On the other hand, edge triggered FFs are bistable elements with synchronous inputs whose state depends on inputs only at the triggering transition of clock pulse. Edge triggering can be positive edge triggering or negative edge triggering.

A latch is constructed using two cross-coupled NOR gates or NAND gates. A latch may be active high latch or an active-low latch.

NOR and NAND Latch circuits are shown in Fig.



2. R-S Flip-Flop- R-S flip-flop has two inputs labelled set (S) and reset (R) and two complementary outputs Q and $\bar{Q}$. R-S flip-flop can be built with NOR gates or NAND gates. The NOR R-S is an active high while the NAND FF is active low.

Circuit diagram for NOR gate R-S flip-flop is shown in Fig. 20.18. Truth Table for NOR gate R-S flip-flop is given in Table.

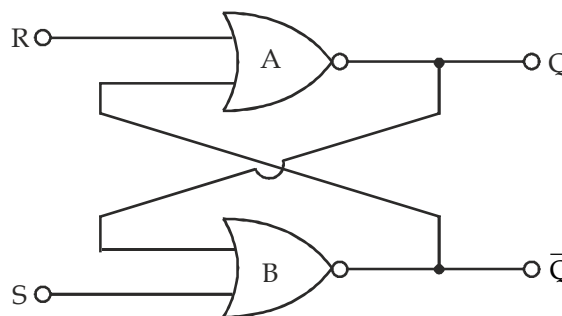


Fig.:- Circuit Diagram of R-S Flip-Flop Using NOR Gates

Table- Truth Table For R-S Flip-Flop

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

3. Clocked R-S Flip-Flop- A clocked R-S latch has a clock input in addition to R and S inputs. A state change can occur only when clock signal becomes high.

Circuit diagram is shown in Fig. and Truth Table is given in Table.

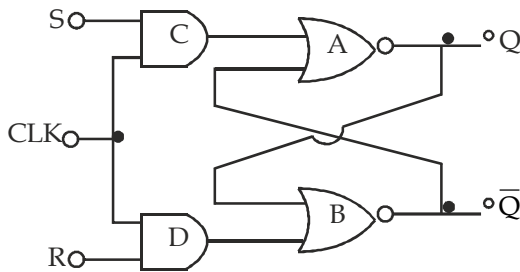


Fig.- Circuit Diagram of Clocked R-S Flip-Flop

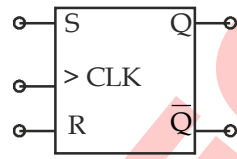
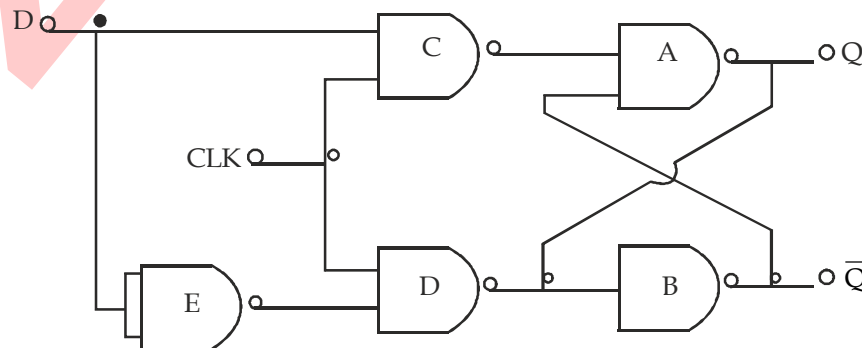


Fig.- Symbol of Clocked R-S Flip-Flop

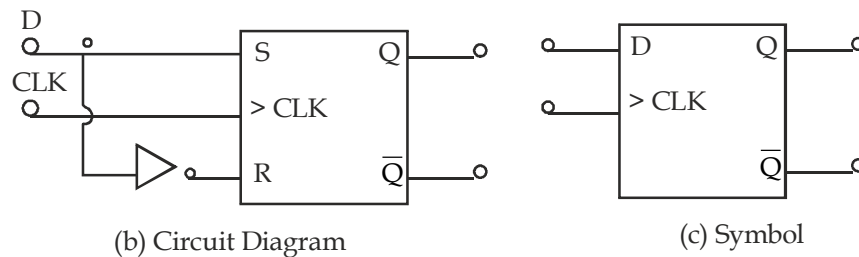
Table- Truth Table For Clocked R-S Flip-Flop

R_n	S_n	CLK	$Q_n + 1$
*	*	0	Q_n
0	0	1	Q_n
1	0	1	0
0	1	1	1
1	1	1	?

4. D Flip-Flop- The D(delay) flip-flop has only a single data input (D) and clock input (CLK). It has a common D input and drives the R-S inputs-S input directly and R input through an inverter. A clocked D FF uses a clock signal to enable and disable the latch. The logic diagram, circuit diagram and symbol for D flip-flop are shown in Figs. 20.21(a), (b) and (c) respectively.



(a) Logic Diagram with NAND Gates



5. J-K Flip-Flop- The functioning of J-K flip-flop is identical to that of the R-S flip-flop in RESET, SET, and no change conditions of operation. The difference is that the J-K flip-flop has no invalid state as does the R-S flip-flop. Therefore, the J-K flip-flop is a very versatile device that finds wide application in digital devices such as counters, registers, arithmetic logic units, and other digital systems.

Master-Slave J-K Flip-Flop. It is made up of two FFs-a master and a slave. Master-slave FFs are called pulse-triggered FFs, because the time duration required for its output to change state is equal to the width of one clock pulse.

6. T-Flip-Flop- A T flip-flop is obtained by connecting the J and K inputs of JK edge triggered FF. In essence the T flip-flop can be treated as a frequency divider or more generally as a device which takes the input frequency at the clock terminal and divides it by 2.

50.

Sol. Digital-To-Analog (D/A) Conversion- Basically, D/A conversion is the process of taking a value represented in digital code (such as simple binary or BCD) and converting it into a voltage or current which is proportional to the digital value.

The basic function of a DAC is shown in given Fig.

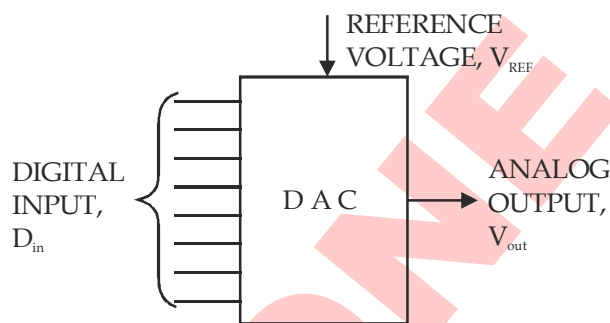


Fig.- Basic Function of an DAC

Digital input D_{in} is defined as an n-bit digital word such that

$$D_{in} = D_0 \times 2^{-n} + D_1 \times 2^{-(n-1)} + D_2 \times 2^{-(n-2)} + \dots D_{n-1} 2^{-1}$$

where D_0 is a binary digit and equals a 1 or a 0. D is also defined as the least significant bit (LSB) and D_{n-1} the most significant bit (MSB). The analog output signal, V_{out} is related to the digital input signal, D_{in} through an analog voltage reference, V_{REF} . The relationship between these three signals is given by

$$V_{out} = V_{REF} (D_0 \times 2^{-n} + D_1 \times 2^{-(n-1)} + D_2 \times 2^{-(n-2)} + \dots D_{n-1} 2^{-1}) = V_{REF} \times D_{in}$$

V_{LSB} may be defined as to be the voltage change when one LSB changes or mathematically

$$V_{LSB} = \frac{V_{REF}}{2^n}$$

The maximum value of output voltage of a DAC is not the reference voltage but rather $V_{REF} (1 - 2^{-n})$ or equivalent to $V_{REF} - V_{LSB}$.

The most commonly used DACs are Weighted-Resistor DAC and Binary Ladder (or R-2R Ladder) DAC.



Weighted-resistor DAC consists of a precision resistor ladder network, a reference precision voltage supply, logic inputs, semiconductor switches and an op-amp. There are two serious drawbacks of the weighted-resistor DAC. The biggest problem is the large difference in resistor values between the LSB and MSB, especially in high-resolution (or many bit) DAC. Secondly, the MSB resistor is required to handle a much greater current than that used for LSB resistor.

Binary ladder or R-2R ladder DAC is the most popular DAC. In general, the voltage resolution for an

n-stage ladder network is given as $\frac{V_{REF}}{2^n}$

51.

Sol. Analog-To-Digital (A/D) Conversion- The basic function of an ADC is shown in Fig. The digital output, D_{out} is the n-bit digital output word while V_{in} , and V_{REF} are the analog input and reference signals.

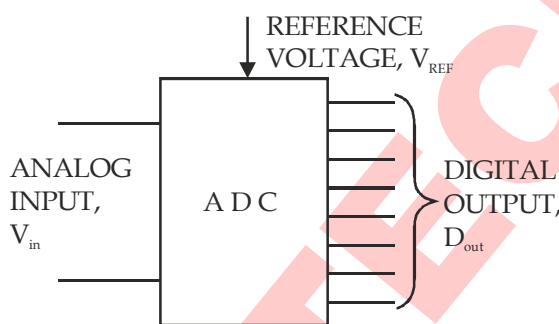


Fig.:- Basic Function of an ADC

For an ADC, these signals are related by equation

$$V_{REF} (D_0 2^{-2} + D_1 \times 2^{-(n-1)} + D_2 \times 2^{-(n-2)} + \dots + D_{n-1} 2^{-1}) = V_{in} \pm V_x$$

where D_0 is a binary digit and equals a 1 or a 0. D_0 is also defined as the least significant bit (LSB) and D_{n-1} is the most significant bit (MSB). V_x is given by the following equation

$$-\left(\frac{1}{2}\right)V_{LSB} \leq V_x \leq \left(\frac{1}{2}\right)V_{LSB}$$

The analog-to-digital conversion process is known as quantization. The noteworthy point is that now there is a range of inputs that provides the same digital output word. There is loss of some information and some distortion is introduced into the signal during the given process. This signal ambiguity results in what is known as quantization error or quantization noise. DAC does not have this effect because the output signals are well defined.

The analog-to-digital (A/D) conversion is the process of converting an analog input voltage into an equivalent digital signal. The operation is somewhat more complex and time-consuming than the D/A conversion.

The different methods used for A/D conversion are: Ramp A/D conversion, Dual slope A/D conversion, Successive-approximation A/D conversion, Voltage-to-frequency A/D conversion etc. Ramp A/D conversion, also known as counter type A/D conversion, is the simplest and popular method of conversion but it is somewhat slower ADC. However, it can digitize high resolution signals.

The averaging characteristics and cancellation of errors that usually limit the performance of a ramp type ADC are the main advantages of dual slope ADC.

Typically, a dual-slope ADC is limited to conversion speeds of from 1,000 to 2,000 samples per second at a resolution of 10 or 12-bits as a result of limitations on the counter counting speed. The conversion rate can be increased to approximate 30,000 samples per second at a 14-bit resolution without significantly increasing the logic requirements by using a two-step integration during the second integration period.



Successive approximation A/D conversion is one of the most widely used method of A/D conversion. Though it employs more complex circuitry than that used by ramp A/D conversion but it has much shorter conversion time. In addition, it has a fixed value of conversion time that does not depend upon the value of the analog input. Successive-approximation ADC can be employed at conversion speeds of up to about 100,000 samples per second at resolutions of up to 16-bits (not including sign). At lower resolutions, speeds of over 250,000 samples per second are practical.

Voltage-to-frequency A/D conversion provides measurement of the true average of the input signal over the ramp duration and so provides high discrimination against noise present at the input. However, the digitizing rates are slow because of long integration durations. The accuracy of this method is comparable with the ramp type ADC and is limited by the stability of the integrator time constant and the stability and accuracy of the comparator.

ZONE TECH