**EE6301 DIGITAL LOGIC CIRCUITS**

**TWO MARK QUESTIONS WITH ANSWERS**

**UNIT-I**

**NUMBERING SYSTEMS AND DIGITAL LOGIC**

**1) What are basic properties of Boolean algebra?**

The basic properties of Boolean algebra are commutative property, associative Property and distributive property.

**2) State the associative property of boolean algebra.**

The associative property of Boolean algebra states that the OR ing of several variables results in the same regardless of the grouping of the variables. The associative property is stated as follows: A+ (B+C) = (A+B) +C

**3) State the commutative property of Boolean algebra**.

The commutative property states that the order in which the variables are OR ed makes no difference. The commutative property is: A+B=B+A

**4) State the distributive property of Boolean algebra.**

The distributive property states that AND ing several variables and OR ing the result With a single variable is equivalent to OR ing the single variable with each of the the several Variables and then AND ing the sums. The distributive property is: A+BC= (A+B) (A+C)

**5) State the absorption law of Boolean algebra**.

The absorption law of Boolean algebra is given by X+XY=X, X(X+Y) =X.

**6) State De Morgan's theorem**.

De Morgan suggested two theorems that form important part of Boolean algebra. They

are,

1. The complement of a product is equal to the sum of the complements. (AB)' = A' + B'
2. The complement of a sum term is equal to the product of the complements. (A + B)' = A'B'

**7) Reduce A (A + B)**

A (A + B) = AA + AB= A (1 + B) [1 + B = 1]= A.

**8) Reduce A'B'C' + A'BC' + A'BC**

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

* + A'C' + A'BC [A + A' = 1]
	+ A'(C' + BC)
	+ A'(C' + B) [A + A'B = A + B]
1. **Reduce AB + (AC)' + AB’C (AB + C)**
* AB + (AC)' + AB'CC [A.A' = 0]
* AB + (AC)' + AB'C [A.A = 1]
* AB + A' + C' =AB'C [(AB)' = A' + B']
* A' + B + C' + AB'C [A + AB' = A + B]
* A' + B'C + B + C' [A + A'B = A + B]
* A' + B + C' + B'C

=A' + B + C' + B'

=A' + C' + 1

* + - 1 [A + 1 =1]
1. **Simplify the following expression Y = (A + B) (A + C’) (B' + C’**)

Y = (A + B) (A + C’) (B' + C’)

* + (AA' + AC +A'B +BC) (B' + C') [A.A' = 0]
	+ (AC + A'B + BC) (B' + C’)
	+ AB'C + ACC' + A'BB' + A'BC' + BB'C + BCC'
	+ AB'C + A'BC'
1. **Show that (X + Y' + XY) (X + Y') (X'Y) = 0**

(X + Y' + XY)(X + Y')(X'Y) = (X + Y' + X) (X + Y’) (X' + Y) [A + A'B = A + B]

* + (X + Y’) (X + Y’) (X'Y) [A + A = 1]
	+ (X + Y’) (X'Y) [A.A = 1]
	+ X.X' + Y'.X'.Y
	+ 0 [A.A' = 0]
1. **Prove that ABC + ABC' + AB'C + A'BC = AB + AC + BC** ABC + ABC' + AB'C + A'BC=AB(C + C') + AB'C + A'BC =AB + AB'C + A'BC

=A(B + B'C) + A'BC =A(B + C) + A'BC =AB + AC + A'BC =B(A + C) + AC =AB + BC + AC

=AB + AC +BC ...Proved

1. **Convert the given expression in canonical SOP form Y = AC + AB + BC** Y = AC + AB + BC

=AC (B + B’) + AB (C + C’) + (A + A') BC

=ABC + ABC' + AB'C + AB'C' + ABC + ABC' + ABC =ABC + ABC' +AB'C + AB'C' [A + A =1]

1. **Define duality property**.

Duality property states that every algebraic expression deducible from the postulates Of Boolean algebra remains valid if the operators and identity elements are interchanged. If the dual of an algebraic expression is desired, we simply interchange OR and AND operators and replace 1's by 0's and 0's by 1's.

1. **Find the complement of the functions F1 = x'yz' + x'y'z and F2 = x (y'z' + yz). By applying De-Morgan's theorem.**

F1' = (x'yz' + x'y'z)' = (x'yz')'(x'y'z)' = (x + y' + z)(x + y +z') F2' = [x (y'z' + yz)]' = x' + (y'z' + yz)'

* + x' + (y'z')'(yz)'x' + (y + z) (y' + z')
1. **Simplify the following expression Y = (A + B) (A = C) (B + C)**
	* (A A + A C + A B + B C) (B + C)
	* (A C + A B + B C) (B + C)
	* A B C + A C C + A B B + A B C + B B C + B C C = A B C

.

**17) Write the maxterms corresponding to the logical expression**

**Y = (A + B + C’) (A + B' + C') (A' + B' + C)**

= (A + B + C’) (A + B' + C') (A' + B' + C)

=M1.M3.M6

= M (1, 3, 6)

**LONG ANSWERS**

1. **Explain binary number to gray code conversion with example.**

 Any binary number can be converted into equivalent Gray code by the following

steps:

(*i*) the MSB of the Gray code is the same as the MSB of the binary number;

(*ii*) the second bit next to the MSB of the Gray code equals the Ex-OR of the MSB and

second bit of the binary number; it will be 0 if there are same binary bits or it will

be 1 for different binary bits;

(*iii*) the third bit for Gray code equals the exclusive-OR of the second and third bits

of the binary number, and similarly all the next lower order bits follow the same

mechanism.

**Example .** *Convert (564)10 into Gray code.*

**Step 1.** Convert the decimal 564 into equivalent binary.

Decimal number 564

Binary number 1000110100

**Step 2.** Convert the binary number into equivalent Gray code.

1  0  0  0 1 1 0  1  0 0 Binary

1 1 0 0 1 0 1 1 1 0 Gray

**Example** *Obtain the canonical sum of product form of the following function.*

*F (A, B, C, D) = AB + ACD*

**Solution.** F (A, B, C, D) = AB + ACD

= AB (C + C′) (D + D′) + ACD (B + B′)

= (ABC + ABC′) (D + D′) + ABCD + AB′CD

= ABCD + ABCD′+ ABC′D + ABC′D′+ ABCD + AB′CD

= ABCD + ABCD′+ ABC′D + ABC′D′+ AB′CD

Hence above is the canonical sum of the product expression of the given function

**Example** *Obtain the canonical product of the sum form of the following function.*

*F (A, B, C) = A + B*′*C*

**Solution.** In the above three-variable expression, the function is given at sum of the

product form. First, the function needs to be changed to product of the sum form by applying

the distributive law as shown below.

F (A, B, C) = A + B′C

= (A + B′) (A + C)

Now, in the above expression, C is missing from the fi rst term and B is missing from

the second term. Hence CC′is to be added with the fi rst term and BB′is to be added with

the second term as shown below.

F (A, B, C) = (A + B′) (A + C)

= (A + B′+ CC′) (A + C + BB′)

= (A + B′+ C) (A + B′+ C′) (A + B + C) (A + B′+ C)

[using the distributive property, as X + YZ = (X + Y) (X + Z)]

= (A + B′+ C) (A + B′+ C′) (A + B + C)

[as (A + B′+ C) (A + B′+ C) = A + B′+ C]

Hence the canonical product of the sum expression for the given function is

F (A, B, C) = (A + B′+ C) (A + B′+ C′) (A + B + C).

**Example 3.17.** *Realize the function F = (A + B)(A’ + C)(B + D) by (i) basic gates,*

*(ii) NAND gates only, (iii) NOR gates only.*

**Solution.** (*i*) The function is realized basic gates as in Figure a



**Figure a**

(*ii*) Realization by NAND gates only is demonstrated in Figure b



**Figure b**

(*iii*) The given function has been implemented with NOR gates only in Figure c.



**Figure c**

****

****

****

****

****

 







UNIT II

GATE LEVEL MINIMIZATION

**1) State the limitations of karnaugh map.**

* 1. Generally it is limited to six variable map (i.e) more then six variable involving expression are not reduced.
	2. The map method is restricted in its capability since they are useful for simplifying only Boolean expression represented in standard form.

**2) What are called don’t care conditions?**

In some logic circuits certain input conditions never occur, therefore the Corresponding output never appears. In such cases the output level is not defined, it can be either high or low. These output levels are indicated by ‘X’ or‘d’ in the truth tables and are called don’t care conditions or incompletely specified functions.

**3) What is a prime implicant?**

A prime implicant is a product term obtained by combining the maximum possible number of adjacent squares in the map.

**4) What is an essential implicant?**

If a min term is covered by only one prime implicant, the prime implicant is said to be Essential.

**5) Write down the steps in implementing a Boolean function with levels of NAND Gates.**

* + - Simplify the function and express it in sum of products.
		- Draw a NAND gate for each product term of the expression that has at least two Literals.
		- The inputs to each NAND gate are the literals of the term.
		- This constitutes a group of first level gates.
		- Draw a single gate using the AND-invert or the invert- OR graphic symbol in the second level, with inputs coming from outputs of first level gates.
		- A term with a single literal requires an inverter in the first level. How ever if the single literal is complemented, it can be connected directly to an input of the second level NAND gate.

**6) Give the general procedure for converting a Boolean expression in to multilevel NAND diagram?**

* + Draw the AND-OR diagram of the Boolean expression.
	+ Convert all AND gates to NAND gates with AND-invert graphic symbols.
	+ Convert all OR gates to NAND gates with invert-OR graphic symbols.
	+ Check all the bubbles in the same diagram. For every bubble that is not compensated by another circle along the same line, insert an inverter or complement the input literal.

**LONG ANSWERS**

**1) Explain about k-map**

Karnaugh maps provide a systematic method to obtain simplified sum-of-products (SOPs) Boolean expressions. This is a compact way of representing a truth table and is a technique that is used to simplify logic expressions. It is ideally suited for four or less variables, becoming cumbersome for five or more variables. Each square represents either a minterm or maxterm. A K-map of n variables will have 2 squares. For a Boolean expression, product terms are denoted by 1's, while sum terms are denoted by 0's - but 0's are often left blank.

A K-map consists of a grid of squares, each square representing one canonical minterm combination of the variables or their inverse. The map is arranged so that squares representing minterms which differ by only one variable are adjacent both vertically and horizontally. Therefore XY'Z' would be adjacent to X'Y'Z' and would also adjacent to XY'Z and XYZ'.

**2) Solve the following expression using 2-variable K-Map**

**F= X'Y+XY+XY'**

****

**F = X + Y**

**3) explain 3 variable K-Map with an example**

There are 8 minterms for 3 variables (X, Y, Z). Therefore, there are 8 cells in a 3-variable K-map. One important thing to note is that K-maps follow the gray code sequence, not the binary one.

****

Using gray code arrangement ensures that minterms of adjacent cells differ by only ONE literal. (Other arrangements which satisfy this criterion may also be used.)

Each cell in a 3-variable K-map has 3 adjacent neighbours. In general, each cell in an n-variable K-map has n adjacent neighbours.

****

There is wrap-around in the K-map

 X'Y'Z' (m0) is adjacent to X'YZ' (m2)

 XY'Z' (m4) is adjacent to XYZ' (m6)

****

**Example**

F = XYZ'+XYZ+X'YZ

****

F = XY + YZ

**4) Solve the following expression using 3 variable K-Map**

F(X,Y,Z) = (1,3,4,5,6,7)

****

F = X + Z

**5) explain 4 variable K-Map with an example**

There are 16 cells in a 4-variable (W, X, Y, Z); K-map as shown in the figure below

****

There are 2 wrap-around: a horizontal wrap-around and a vertical wrap-around. Every cell thus has 4 neighbours. For example, the cell corresponding to minterm m0 has neighbours m1, m2, m4 and m8.

****

**Example**

F(W,X,Y,Z) = (1,5,12,13)

****

F = WY'Z + W'Y'Z

6) solve the following expression using 4 variable K-Map

F(W,X,Y,Z) = (4, 5, 10, 11, 14, 15)

****

F = W'XY' + WY

**7) simplify the Boolean function** F(w,x,y, z) = ∑(1,3,7,11,15), dc(w,x,y,z) = ∑(0,2,5)

Sol:

****

**And also the same can be solved in the following manner**

****

**8) Implement all the gates by using universal gates.**

NAND and NOR gates are called universal gates as any digital function can be implemented by using only NAND or NOR gate alone. Combinational and sequential both type of circuits can be implemented using NAND or NOR gate. Let’s see how we can derive all basic gates from NAND:



Let’s now see that how we can derive all basic gates from NOR gate:



**9) Implement 2 variable XOR gate using NAND only in minimum number of gates.**

F= x XOR y = x’y+xy’ = x’y+xy’+xx’+yy’ = (x+y) (x’+y’)

Now we need to implement this circuit using NAND gates

F= (x+y)(xy)’ = x. (xy)’ + y. (xy)’

Take compliment

F’= ( x. (xy)’ + y. (xy)’ )’ = (x. (xy)’)’. (y. (xy)’)

Take compliment again

F=( (x. (xy)’)’. (y. (xy)’) )’

Now we can implement this using NAND gates



So we get that we need minimum of 4 NAND gates to implement XOR gate and if we are to implement XNOR gate then we’ll use 5 NAND gates with 5th gate used as inverter and placed in-front of 4th NAND gate in above circuit.

**10) Implement 2 variable XNOR gate using NOR only in minimum number of gates.**

F= x XNOR y = (x XOR y)’

F’= (x XOR y) = x’y+xy’ = x’y+xy’+xx’+yy’ = (x+y) (x’+y’)

**F’=** x’(x+y) + y’(x+y)

Take compliment

F= (x’(x+y) + y’(x+y))’ = (x’(x+y) )’ . (y’(x+y))’ =(x+(x+y)’). (y + (x+y)’)

Take compliment again

F’= ( (x+(x+y)’). **(**y + (x+y)’) )’ = (x+(x+y)’)’ + (y + (x+y)’)’

Take compliment again

F= [ (x+(x+y)’)’ + (y + (x+y)’)’ ]’

Now we can implement the circuit of XNOR gate using NOR gates



**Hence we need only 4 NOR GATES to implement XNOR gate and we require 5 NOR gates to implement XOR gate with 5th NOR gate used as inverter and placed in front of 4th NOR gate.**

**UNIT 3**

**COMBINATIONAL CIRCUITS**

**1) Define combinational logic.**

When logic gates are connected together to produce a specified output for certain specified combinations of input variables, with no storage involved, the resulting circuit is called combinational logic.

2) **Write the design procedure for combinational circuits.**

* + The problem definition
	+ Determine the number of available input variables & required O/P variables.
	+ Assigning letter symbols to I/O variables
	+ Obtain simplified Boolean expression for each O/P.
	+ Obtain the logic diagram.

**3) Define half adder and full adder.**

The logic circuit that performs the addition of two bits is a half adder. The circuit that Performs the addition of three bits is a full adder.

**4) Define Decoder.**

A decoder is a multiple - input multiple output logic circuit that converts coded inputs into coded outputs where the input and output codes are different.

**5) What is binary decoder?**

A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2n out puts lines.

**6) Define Encoder.**

An encoder has 2n input lines and n output lines. In encoder the output lines generate the binary code corresponding to the input value.

**7) What is priority Encoder?**

A priority encoder is an encoder circuit that includes the priority function. In priority encoder, if 2 or more inputs are equal to 1 at the same time, the input having the highest priority will take precedence.

**8) Define multiplexer.**

Multiplexer is a digital switch. If allows digital information from several sources to be routed onto a single output line.

**9) What do you mean by comparator?**

A comparator is a special combinational circuit designed primarily to compare the relative magnitude of two binary numbers.

**LONG ANSWERS**

**1) What is combinational circuit**

Combinatorial Circuits are circuits which can be considered to have the following generic structure.



Whenever the same set of inputs is fed in to a combinatorial circuit, the same outputs will be generated. Such circuits are said to be stateless. Some simple combinational logic elements that we have seen in previous sections are "Gates".



All the gates in the above figure have 2 inputs and one output; combinational elements simplest form are "not" gate and "buffer" as shown in the figure below. They have only one input and one output.



**2)What is Decoders explain with an example**

Enable inputs must be on for the decoder to function, otherwise its outputs assume a single "disabled" output code word.

Decoding is necessary in applications such as data multiplexing, 7 segment display and memory address decoding. Figure below shows the pseudo block of a decoder.



**Binary n-to-2n Decoders**

A binary decoder has n inputs and 2n outputs. Only one output is active at any one time, corresponding to the input value. Figure below shows a representation of Binary n-to-2n decoder



**2-to-4 Binary Decoder**

A 2 to 4 decoder consists of two inputs and four outputs, truth table and symbols of which is shown below.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **X**  | **Y**  | **F0**  | **F1**  | **F2**  | **F3**  |
| 0  | 0  | 1  | 0  | 0  | 0  |
| 0  | 1  | 0  | 1  | 0  | 0  |
| 1  | 0  | 0  | 0  | 1  | 0  |
| 1  | 1  | 0  | 0  | 0  | 1  |

**Symbol**



To minimize the above truth table we may use kmap, but doing that you will realize that it is a waste of time. One can directly write down the function for each of the outputs. Thus we can draw the circuit as shown in figure below.

**Note:** Each output is a 2-variable minterm (X'Y', X'Y, XY', XY)

|  |
| --- |
| **Circuit Example-** |



**3) 3-to-8 Binary Decoder**

A 3 to 8 decoder consists of three inputs and eight outputs, truth table and symbols of which is shown below.

**Truth table**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **X**  | **Y**  | **Z**  | **F0**  | **F1**  | **F2**  | **F3**  | **F4**  | **F5**  | **F6**  | **F7**  |
| 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  |
|  |  |  |  |  |  |  |  |  |  |  |



From the truth table we can draw the circuit diagram as shown in figure below.



**4) Explain about Full adder with its hardware implementation.**

S(x, y, z) = (1,2,4,7)

C(x, y, z) = (3,5,6,7)

**Truth Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **X**  | **Y**  | **Z**  | **C**  | **S**  |
| 0  | 0  | 0  | 0  | 0  |
| 0  | 0  | 1  | 0  | 1  |
| 0  | 1  | 0  | 0  | 1  |
| 0  | 1  | 1  | 1  | 0  |
| 1  | 0  | 0  | 0  | 1  |
| 1  | 0  | 1  | 1  | 0  |
| 1  | 1  | 0  | 1  | 0  |
| 1  | 1  | 1  | 1  | 1  |

From the truth table we know the values for which the sum (s) is active and also the carry (c) is active. Thus we have the equation as shown above and a circuit can be drawn as shown below from the equation derived.



**5) 4to3 Priority Encoder**

The truth table of a 4-input priority encoder is as shown below. The input D3 has the highest priority, D2 has next highest priority, D0 has the lowest priority. This means output Y2 and Y1 are 0 only when none of the inputs D1, D2, D3 are high and only D0 is high.

A 4 to 3 encoder consists of four inputs and three outputs, truth table and symbols of which is shown below.

**Truth Table**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **D3**  | **D2**  | **D1**  | **D0**  | **Y2**  | **Y1**  | **Y0**  |
| 0  | 0  | 0  | 0  | 0  | 0  | 0  |
| 0  | 0  | 0  | 1  | 0  | 0  | 1  |
| 0  | 0  | 1  | x  | 0  | 1  | 0  |
| 0  | 1  | x  | x  | 0  | 1  | 1  |
| 1  | x  | x  | x  | 1  | 0  | 0  |

Now that we have the truth table, we can draw the Kmaps as shown below.

**Kmaps**

From the Kmap we can draw the circuit as shown below. For Y2, we connect directly to D3.



We can apply the same logic to get higher order priority encoders.

**6) explain about Multiplexer with an example.**

A multiplexer (MUX) is a digital switch which connects data from one of n sources to the output. A number of select inputs determine which data source is connected to the output. The block diagram of MUX with n data sources of b bits wide and s bits wide select line is shown in below figure.



MUX acts like a digitally controlled multi-position switch where the binary code applied to the select inputs controls the input source that will be switched on to the output as shown in the figure below. At any given point of time only one input gets selected and is connected to output, based on the select input signal.

**8-to-1 multiplexer from Smaller MUX**

**Truth Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **S2**  | **S1**  | **S0**  | **F**  |
| 0  | 0  | 0  | I0  |
| 0  | 0  | 1  | I1  |
| 0  | 1  | 0  | I2  |
| 0  | 1  | 1  | I3  |
| 1  | 0  | 0  | I4  |
| 1  | 0  | 1  | I5  |
| 1  | 1  | 0  | I6  |

**Circuit**



**7) 3-variable Function Using 4-to-1 mux**

Implement the function F(X,Y,Z) = S(0,1,3,6) using a single 4-to-1 mux and an inverter. We choose the two most significant inputs X, Y as mux select lines.

Construct truth table.

**Truth Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Select i**  | **X**  | **Y**  | **Z**  | **F**  | **Mux Input i**  |
| **0**  | **0**  | **0**  | **0**  | **1**  | **1**  |
| **0**  | **0**  | **0**  | **1**  | **1**  | **1**  |
| **1**  | **0**  | **1**  | **0**  | **0**  | **Z**  |
| **1**  | **0**  | **1**  | **1**  | **1**  | **Z**  |
| **2**  | **1**  | **0**  | **0**  | **0**  | **0**  |
| **2**  | **1**  | **0**  | **1**  | **0**  | **0**  |
| **3**  | **1**  | **1**  | **0**  | **1**  | **Z'**  |
| **3**  | **1**  | **1**  | **1**  | **0**  | **Z'**  |

**Circuit**



We determine multiplexer input line i values by comparing the remaining input variable Z and the function F for the corresponding selection lines value i

 when XY=00 the function F is 1 (for both Z=0, Z=1) thus mux input0 = 1

 when XY=01 the function F is Z thus mux input1 = Z

 when XY=10 the function F is 0 (for both Z=0, Z=1) thus mux input2 = 0

 when XY=11 the function F is Z' thus mux input3 = Z'

**UNIT 4**

**SEQUENTIAL CIRCUITS**

**1. What are the classifications of sequential circuits?**

The sequential circuits are classified on the basis of timing of their signals into two types. They are,

1) Synchronous sequential circuit. 2) Asynchronous sequential circuit.

**2. Define Flip flop.**

The basic unit for storage is flip flop. A flip-flop maintains its output state either at 1 or 0 until directed by an input signal to change its state.

**3. What are the different types of flip-flop?**

There are various types of flip flops. Some of them are mentioned below they are,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| (1) | RS flip-flop | (2) | SR flip-flop | (3) D flip-flop |
| (4) JK flip-flop | (5) | T flip-flop |  |

1. **What is the operation of RS flip-flop?**
	* When R input is low and S input is high the Q output of flip-flop is set.
	* When R input is high and S input is low the Q output of flip-flop is reset.
	* When both the inputs R and S are low the output does not change.
	* When both the inputs R and S are high the output is unpredictable.
2. **What is the operation of SR flip-flop?**
	* When R input is low and S input is high the Q output of flip-flop is set.
	* When R input is high and S input is low the Q output of flip-flop is reset.
	* When both the inputs R and S are low the output does not change.
	* When both the inputs R and S are high the output is unpredictable.
3. **What is the operation of D flip-flop?**

In D flip-flop during the occurrence of clock pulse if D=1, the output Q is set and if D=0, the output is reset.

1. **What is the operation of JK flip-flop?**
	* When K input is low and J input is high the Q output of flip-flop is set.
	* When K input is high and J input is low the Q output of flip-flop is reset.
	* When both the inputs K and J are low the output does not change
	* When both the inputs K and J are high it is possible to set or reset the Flip-flop (ie) the output toggle on the next positive clock edge.
2. **What is the operation of T flip-flop?**

T flip-flop is also known as Toggle flip-flop.

* + When T=0 there is no change in the output.
	+ When T=1 the output switch to the complement state (ie) the output toggles.
1. **Define race around condition.**

In JK flip-flop output is fed back to the input. Therefore change in the output results change in the input. Due to this in the positive half of the clock pulse if both J and K are high then output toggles continuously. This condition is called race around condition’.

**10. What is edge-triggered flip-flop?**

The problem of race around condition can solved by edge triggering flip flop. The term edge triggering means that the flip-flop changes state either at the positive edge or negative edge of the clock pulse and it is sensitive to its inputs only at this transition of the clock.

**11. What is a master-slave flip-flop?**

A master-slave flip-flop consists of two flip-flops where one circuit serves as a master and the other as a slave.

**12. Explain the flip-flop excitation tables for RS FF.**

In RS flip-flop there are four possible transitions from the present state to the next state. They are,

* + \_ 0\_0 transition: This can happen either when R=S=0 or when R=1 and S=0.
	+ \_ 0\_1 transition: This can happen only when S=1 and R=0.
	+ \_ 1\_0 transition: This can happen only when S=0 and R=1.
	+ \_ 1\_1 transition: This can happen either when S=1 and R=0 or S=0 and R=0.
1. **Explain the flip-flop excitation tables for JK flip-flop**

In JK flip-flop also there are four possible transitions from present state to next state.They

are,

* + \_ 0\_0 transition: This can happen when J=0 and K=1 or K=0.
	+ \_ 0\_1 transition: This can happen either when J=1 and K=0 or when J=K=1.
	+ \_ 1\_0 transition: This can happen either when J=0 and K=1 or when J=K=1.
	+ \_ 1\_1 transition: This can happen when K=0 and J=0 or J=1.
1. **Explain the flip-flop excitation tables for D flip-flop**

In D flip-flop the next state is always equal to the D input and it is independent of the present state. Therefore D must be 0 if Qn+1 has to 0, and if Qn+1 has to be 1 regardless the value of Qn.

**15. Explain the flip-flop excitation tables for T flip-flop**

When input T=1 the state of the flip-flop is complemented; when T=0, the state of the Flip-flop remains unchanged. Therefore, for 0\_0 and 1\_1 transitions T must be 0 and for 0\_1 and 1\_0 transitions must be 1.

**16. Define sequential circuit.**

In sequential circuits the output variables dependent not only on the present input variables but they also depend up on the past history of these input variables.

1. **Give the comparison between combinational circuits and sequential circuits.** Combinational circuits Sequential circuits Memory unit is not required Memory unity is

Required Parallel adder is a combinational circuit Serial adder is a sequential circuit.

**18. What do you mean by present state?**

The information stored in the memory elements at any given time define.s the present state of the sequential circuit.

**19. What do you mean by next state?**

The present state and the external inputs determine the outputs and the next state of the sequential circuit.

**20. State the types of sequential circuits?**

1. Synchronous sequential circuits 2. Asynchronous sequential circuits

**21. Define synchronous sequential circuit**

In synchronous sequential circuits, signals can affect the memory elements only at discrete instant of time.

**22. Define Asynchronous sequential circuit?**

In asynchronous sequential circuits change in input signals can affect memory element at any instant of time.

1. **Give the comparison between synchronous & Asynchronous sequential circuits?** Synchronous sequential circuits Asynchronous sequential circuits. Memory elements are

locked flip-flops Memory elements are either unlocked flip - flops or time delay elements.

**24. What is race around condition?**

In the JK latch, the output is feedback to the input, and therefore changes in the output results change in the input. Due to this in the positive half of the clock pulse if J and K are both high then output toggles continuously. This condition is known as race around condition.

1. **Give the comparison between synchronous & Asynchronous counters. Asynchronous counters**
	* In this type of counter flip-flops are Connected in such a way that output of 1st Flip-flop drives the clock for the next Flipflop
	* All the flip-flops are not clocked Simultaneously

**Synchronous counters**

* In this type there is no connection between output of first flip-flop and clock input of

the next flip – flop

* All the flip-flops are clocked simultaneously

**LONG ANSWERS**

**1) What is sequential circuit**

Digital electronics is classified into combinational logic and sequential logic. Combinational logic output depends on the inputs levels, whereas sequential logic output depends on stored levels and also the input levels.



The memory elements are devices capable of storing binary info. The binary info stored in the memory elements at any given time defines the state of the sequential circuit. The input and the present state of the memory element determines the output. Memory elements next state is also a function of external inputs and present state. A sequential circuit is specified by a time sequence of inputs, outputs, and internal states.

**2) What are the different types of sequential circuits and explain.**

There are two types of sequential circuits. Their classification depends on the timing of their signals:

 Synchronous sequential circuits

 Asynchronous sequential circuits

**Asynchronous sequential circuit**

This is a system whose outputs depend upon the order in which its input variables change and can be affected at any instant of time.

Gate-type asynchronous systems are basically combinational circuits with feedback paths. Because of the feedback among logic gates, the system may, at times, become unstable. Consequently they are not often used.



**Synchronous sequential circuits**

This type of system uses storage elements called flip-flops that are employed to change their binary value only at discrete instants of time. Synchronous sequential circuits use logic gates and flip-flop storage devices. Sequential circuits have a clock signal as one of their inputs. All state transitions in such circuits occur only when the clock value is either 0 or 1 or happen at the rising or falling edges of the clock depending on the type of memory elements used in the circuit. Synchronization is achieved by a timing device called a clock pulse generator. Clock pulses are distributed throughout the system in such a way that the flip-flops are affected only with the arrival of the synchronization pulse. Synchronous sequential circuits that use clock pulses in the inputs are called clocked-sequential circuits. They are stable and their timing can easily be broken down into independent discrete steps, each of which is considered separately.



A clock signal is a periodic square wave that indefinitely switches from 0 to 1 and from 1 to 0 at fixed intervals. Clock cycle time or clock period: the time interval between two consecutive rising or falling edges of the clock.

Clock Frequency = 1 / clock cycle time (measured in cycles per second or Hz)

**3) What is mean by latches and flip flops explain sr latch**

Latches and Flip-flops are one and the same with a slight variation: Latches have level sensitive control signal input and Flip-flops have edge sensitive control signal input. Flip-flops and latches which use this control signals are called synchronous circuits. So if they don't use clock inputs, then they are called asynchronous circuits.

**RS Latch**

RS latch have two inputs, S and R. S is called set and R is called reset. The S input is used to produce HIGH on Q ( i.e. store binary 1 in flip-flop). The R input is used to produce LOW on Q (i.e. store binary 0 in flip-flop). Q' is Q complementary output, so it always holds the opposite value of Q. The output of the S-R latch depends on current as well as previous inputs or state, and its state (value stored) can change as soon as its inputs change. The circuit and the truth table of RS latch is shown below. (This circuit is as we saw in the last page, but arranged to look beautiful



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S**  |  **R**  |  **Q**  |  **Q+**  |   |
|  0  |  0  |  0  |  0  |   |
|  0  |  0  |  1  |  1  |   |
|  0  |  1  |  X  |  0  |   |
|  1  |  0  |  X  |  1  |   |
|  1  |  1  |  X  |  0  |   |

The operation has to be analyzed with the 4 inputs combinations together with the 2 possible previous states.

 **When S = 0 and R = 0:** If we assume Q = 1 and Q' = 0 as initial condition, then output Q after input is applied would be Q = (R + Q')' = 1 and Q' = (S + Q)' = 0. Assuming Q = 0 and Q' = 1 as initial condition, then output Q after the input applied would be Q = (R + Q')' = 0 and Q' = (S + Q)' = 1. So it is clear that when both S and R inputs are LOW, the output is retained as before the application of inputs. (i.e. there is no state change).

 **When S = 1 and R = 0:** If we assume Q = 1 and Q' = 0 as initial condition, then output Q after input is applied would be Q = (R + Q')' = 1 and Q' = (S + Q)' = 0. Assuming Q = 0 and Q' = 1 as initial condition, then output Q after the input applied would be Q = (R + Q')' = 1 and Q' = (S + Q)' = 0. So in simple words when S is HIGH and R is LOW, output Q is HIGH.

 **When S = 0 and R = 1:** If we assume Q = 1 and Q' = 0 as initial condition, then output Q after input is applied would be Q = (R + Q')' = 0 and Q' = (S + Q)' = 1. Assuming Q = 0 and Q' = 1 as initial condition, then output Q after the input applied would be Q = (R + Q')' = 0 and Q' = (S + Q)' = 1. So in simple words when S is LOW and R is HIGH, output Q is LOW.

 **When S = 1 and R =1 :** No matter what state Q and Q' are in, application of 1 at input of NOR gate always results in 0 at output of NOR gate, which results in both Q and Q' set to LOW (i.e. Q = Q'). LOW in both the outputs basically is wrong, so this case is invalid.

The waveform below shows the operation of NOR gates based RS Latch.



**4) Setup and Hold Time**

For synchronous flip-flops, we have special requirements for the inputs with respect to clock signal input. They are

 **Setup Time:** Minimum time period during which data must be stable before the clock makes a valid transition. For example, for a posedge triggered flip-flop, with a setup time of 2 ns, Input Data (i.e. R and S in the case of RS flip-flop) should be stable for at least 2 ns before clock makes transition from 0 to 1.

 **Hold Time:** Minimum time period during which data must be stable after the clock has made a valid transition. For example, for a posedge triggered flip-flop, with a hold time of 1 ns. Input Data (i.e. R and S in the case of RS flip-flop) should be stable for at least 1 ns after clock has made transition from 0 to 1.

 If data makes transition within this setup window and before the hold window, then the flip-flop output is not predictable, and flip-flop enters what is known as **meta stable state**. In this state flip-flop output oscillates between 0 and 1. It takes some time for the flip-flop to settle down. The whole process is called **metastability**. You could refer to tidbits section to know more information on this topic.

The waveform below shows input S (R is not shown), and CLK and output Q (Q' is not shown) for a SR posedge flip-flop.



**5) D Latch**

The RS latch seen earlier contains ambiguous state; to eliminate this condition we can ensure that S and R are never equal. This is done by connecting S and R together with an inverter. Thus we have D Latch: the same as the RS latch, with the only difference that there is only one input, instead of two (R and S). This input is called D or Data input. D latch is called D transparent latch for the reasons explained earlier. Delay flip-flop or delay latch is another name used. Below is the truth table and circuit of D latch.

In real world designs (ASIC/FPGA Designs) only D latches/Flip-Flops are used.



|  |  |  |
| --- | --- | --- |
| **D**  | **Q**  | **Q+**  |
| 1  | X  | 1  |
| 0  | X  | 0  |

Below is the D latch waveform, which is similar to the RS latch one, but with R removed.



**6) JK Master Slave Flip-Flop**

All sequential circuits that we have seen in the last few pages have a problem (All level sensitive sequential circuits have this problem). Before the enable input changes state from HIGH to LOW (assuming HIGH is ON and LOW is OFF state), if inputs changes, then another state transition occurs for the same enable pulse. This sort of multiple transition problem is called racing.

If we make the sequential element sensitive to edges, instead of levels, we can overcome this problem, as input is evaluated only during enable/clock edges.



In the figure above there are two latches, the first latch on the left is called master latch and the one on the right is called slave latch. Master latch is positively clocked and slave latch is negatively clocked.



**7) What are the difference between latches and flip flop**



**3**

**UNIT 5**

**MEMORY**

**1. Explain ROM.**

A read only memory (ROM) is a device that includes both the decoder and the OR gates within a single IC package. It consists of n input lines and m output lines. Each bit Combination of the input variables is called an address. Each bit combination that comes out of the output lines is called a word. The number of distinct addresses possible with n input variables is 2n.

**2. What are the types of ROM?**

1. PROM 2. EPROM 3. EEPROM

**3. Explain PROM.**

PROM (Programmable Read Only Memory) it allows user to store data or program. PROMs use the fuses with materiallike nichrome and polycrystalline. The user can blow these fuses by passing around 20 to 50 mA of current for the period 5 to 20ìs.The blowing of fuses is called programming of ROM. The PROMs are one time programmable. Once programmed, the information is stored permanent.

**4. Explain EPROM.**

EPROM (Erasable Programmable Read Only Memory) EPROM use MOS circuitry. They store 1’s and 0’s as a packet of charge in a buried layer of the IC chip. We can erase the stored data in the EPROMs by exposing the chip to ultraviolet light via its quartz window for 15 to 20 minutes. It is not possible to erase selective information. The chip can be reprogrammed.

**5. Explain EEPROM**.

EEPROM (Electrically Erasable Programmable Read Only Memory). EEPROM also use MOS circuitry. Data is stored as charge or no charge on an insulated layer or an insulated floating gate in the device. EEPROM allows selective erasing at the register level rather than erasing all the information since the information can be changed by using electrical signals.

6**. Define address and word:**

In a ROM, each bit combination of the input variable is called on address. Each bit combination that comes out of the output lines is called a word.

**7. What are the types of ROM.?**

1. Masked ROM.

2. Programmable Read only Memory

* 1. Erasable Programmable Read only memory.
	2. Electrically Erasable Programmable Read only Memory.
1. **What is programmable logic array? How it differs from ROM?**

In some cases the number of don’t care conditions is excessive, it is more economical to use a second type of LSI component called a PLA. A PLA is similar to a ROM in concept; however it does not provide full decoding of the variables and does not generates all the min terms as in the ROM.

**9. What is mask - programmable?**

With a mask programmable PLA, the user must submit a PLA program table to the manufacturer.

**10. What is field programmable logic array?**

The second type of PLA is called a field programmable logic array. The user by means of certain recommended procedures can program the EPLA.

**11. List the major differences between PLA and PAL**

**PLA:**Both AND and OR arrays are programmable and Complex Costlier than PAL

**PAL:**AND arrays are programmable OR arrays are fixed Cheaper and Simpler

**12. Define PLD.**

Programmable Logic Devices consist of a large array of AND gates and OR gates that Can be programmed to achieve specific logic functions.

**13. Give the classification of PLDs.**

PLDs are classified as PROM (Programmable Read Only Memory), Programmable Logic Array (PLA), Programmable Array Logic (PAL), and Generic Array Logic (GAL)

**14. Define PROM.**

PROM is Programmable Read Only Memory. It consists of a set of fixed AND gates Connected to a decoder and a programmable OR array.

**15. Define PLA.**

PLA is Programmable Logic Array (PLA). The PLA is a PLD that consists of a Programmable AND array and a programmable OR array.

**16. Define PAL.**

PAL is Programmable Array Logic. PAL consists of a programmable AND array and a fixed OR array with output logic.

**17. Why was PAL developed?**

It is a PLD that was developed to overcome certain disadvantages of PLA, such as longer delays due to additional fusible links that result from using two programmable arrays and more circuit complexity.

**18. Define GAL.**

GAL is Generic Array Logic. GAL consists of a programmable AND array and a fixed OR array with output logic.

**19. Why the input variables to a PAL are buffered**

The input variables to a PAL are buffered to prevent loading by the large number of AND gate inputs to which available or its complement can be connected.

1. **What does PAL 10L8 specify?**

PAL - Programmable Logic Array 10 - Ten inputs

L - Active LOW Ouput 8 - Eight Outputs

1. **What is CPLD?**

CPLDs are Complex Programmable Logic Devices. They are larger versions of PLDs with a centralized internal interconnect matrix used to connect the device macro cells together.

**22. Define bit, byte and word.**

The smallest unit of binary data is bit. Data are handled in a 8 bit unit called byte. A complete unit of information is called a word which consists of one or more bytes.

**23. How many words can a 16x8 memory can store?**

A 16x8 memory can store 16,384 words of eight bits each

**24. Define address of a memory.**

The location of a unit of data in a memory is called address.

**25. What is Read and Write operation?**

The Write operation stores data into a specified address into the memory and the Read operation takes data out of a specified address in the memory.

**26. Why RAMs are called as Volatile?**

RAMs are called as Volatile memories because RAMs lose stored data when the power is turned OFF.

**27. Define ROM**.

ROM is a type of memory in which data are stored permanently or semi permanently.

Data can be read from a ROM, but there is no write operation.

**28. Define RAM.**

RAM is Random Access Memory. It is a random access read/write memory. The data can be read or written into from any selected address in any sequence.

**29. Define Static RAM and dynamic RAM.**

Static RAM use flip flops as storage elements and therefore store data indefinitely as long as dc power is applied. Dynamic RAMs use capacitors as storage elements and cannot retain data very long without capacitors being recharged by a process called refreshing.

**30. List the two types of SRAM.**

Asynchronous SRAMs and Synhronous Burst SRAMs

**31. List the basic types of DRAMs.**

Fast Page Mode DRAM,Extended Data Out DRAM(EDO DRAM),Burst EDO DRAM and Synchronous DRAM.

**32. Define a bus.**

A bus is a set of conductive paths that serve to interconnect two or more functional components of a system or several diverse systems.

**33. Define Cache memory.**

It is a relatively small, high-speed memory that can store the most recently used instructions or data from larger but slower main memory.

**34. What is the technique adopted by DRAMs.**

DRAMs use a technique called address multiplexing to reduce the number of address

lines.

**35.Give the feature of UV EPROM.**

UV EPROM is electrically programmable by the user, but the store data must be erased by exposure to ultra violet light over a period of several minutes.

**36. Give the feature of flash memory**.

The ideal memory has high storage capacity, non-volatility; in-system read and write capability, comparatively fast operation. The traditional memory technologies such as ROM, PROM, EEPROM individually exhibits one of these characteristics, but no single technology has all of them except the flash memory.

**37. What are Flash memories?**

They are high density read/write memories that are non-volatile, which means data can be stored indefinitely with out power.

1. **List the three major operations in a flash memory.** Programming, Read and Erase operation
2. **What is a FIFO memory?**

The term FIFO refers to the basic operation of this type of memory in which the first data bit written into the memory is to first to be read out.

1. **List basic types of programmable logic devices**.
	1. Read only memory 2. Programmable logic Array 3. Programmable Array Logic

**LONG ANSWERS**

**1) IMPLEMENT THE FOLLOWING BY USING PLA**

****

**2) IMPLEMENT FULL ADDER USING PAL**

****

**3) difference between RAM and ROM**

|  | **ROM** |
| --- | --- |
| **1.** | Stands for *Randon-access Memory* | Stands for *Read-only memory* |
| **2.** | RAM is a read and write memory | Normally ROM is read only memory and it can not be overwritten. However, EPROMs can be reprogrammed |
| **3.** | RAM is faster | ROM is relatively slower than RAM |
| **4.** | RAM is a **volatile memory**. It means that the data in RAM will be lost if power supply is cut-off | ROM is permanent memory. Data in ROM will stay as it is even if we remove the power-supply |
| **5.** | There are mainly two types of RAM; *static RAM* and *Dynamic RAM* | There are several types of ROM; Erasable ROM, Programmable ROM, EPROM etc. |
| **6.** | RAM stores all the applications and data when the computer is up and running | ROM usually stores instructions that are required for starting (booting) the computer |
| **7.** | Price of RAM is comparatively high | ROM chips are comparatively cheaper |
| **8.** | RAM chips are bigger in size | ROM chips are smaller in size |
| **9.** | Processor can directly access the content of RAM | Content of ROM are usually first transferred to RAM and then accessed by processor. This is done in order to be able to access ROM content at a faster speed. |
| **10.** | RAM is often installed with large storage. | Storage capacity of ROM installed in a computer is much lesser than RAM |